APPENDIX 2 - ITEMS 1-11

p. 128

Appendix 2 · Item 1

Transactions Papers

Use of Residual Redundancy in the Design of Joint Source/Channel Coders

Khalid Sayood, *Member, IEEE,* and Jay C. Borkenhagen, *Member, IEEE*

Abstract— **The need to transmit large amounts of data over a band-limited channel has led to the development of various data compression schemes. Many of these schemes function by attempting to remove redundancy from the data stream. An unwanted side-effect of this approach is to make the information transfer process more vulnerable to channel noise. Efforts at protecting against errors involve the reinsertion of redundancy and an increase in bandwidth requirements. We present a technique for providing error protection without the additional overhead required for channel coding. We start from the premise that, during source coder design, for the sake of simplicity or due to imperfect knowledge, assumptions have to be made about the source which are often incorrect. This results in residual redundancy at the output of the source coder. The residual redundancy can then be used to provide error protection in much the same way as the insertion of redundancy in convolutional coding provides error protection. In this paper we develop an approach for utilizing this redundancy. To show the validity of this approach, we apply it to image coding using DPCM, and obtain substantial performance gains, both in terms of objective as well as subjective measures.**

I. Introduction

THE utilization of redundancy removal techniques is motivated by the need for more efficient channel utilization for data transmission and reduced memory requirements for data storage. The source coder removes redundancy from the data, and this redundancy is later reinserted at the receiver. The design of the source coder is, in general, based solely on the source statistics. The removal of redundancy makes the transmitted data especially vulnerable to channel noise. To combat the effect of channel noise, channel coding is used, which entails the controlled insertion of redundancy into the transmitted data [1]. The channel coding procedures are designed without any reference to the source characteristics. This approach is justified in some sense by an important result of Shannon [2] which shows that the source and channel coding operations can be separated without any loss of optimality. However, in Shannon's work there is no constraint on the complexity of the coders involved. In practical systems, where there are limits on complexity, this separation may not be possible [3].

Shannon showed that for systems transmitting at a rate below the capacity of the channel, essentially error-free transmission can be attained. If the link between the source coder and decoder is error-free, there is no need for the effect of errors to be taken into account in the design of the source coder/decoder pair since there are no errors. However, most communication links are not error-free, and the effect of these errors has to be studied and if possible mitigated. Modestino, Daut, and Vickers [4] in their study of transform coding of images for transmission over noisy channels have shown that for moderate input signal-to-noise ratios, the use of a greater average number of bits per coefficient actually degrades the performance of the system, and reduces the output signal-to-noise ratio. To combat this effect, they examine tradeoffs between allocating bits for source or channel coding. Comstock and Gibson [5] extend this work and provide an explicit mechanism for allocating bits between the source coder and a Hamming channel coder. A similar situation is evident for DPCM systems. As can be seen from Fig. 1, while the higher rate DPCM system performs better under relatively noiseless conditions, its performance drops below that of the lower rate system under noisier conditions. Thus, under very noisy channel conditions, the DPCM system that provides the lowest source coding distortion turns out to have the highest overall distortion. Chang and Donaldson [6] present an analysis of the DPCM system operating under noisy channel conditions. Examining the case where no separate channel coding is being used, they demonstrate the usefulness of incorporating both channel and source statistics in the design of the source coder.

It seems clear from the above that there is a need to consider the effect of channel errors when designing source coders. An early effort in this direction was the difference detection and correction scheme of Steele, Goodman, and McGonegal [7], [8] for broadcast-quality speech. In this scheme the receiver infers an error has occurred whenever an individual sample-

Paper approved by the Editor for Quantization and Speech/Image Coding of the IEEE Communications Society. Manuscript received April 30, 1987; revised August 8, 1990. This work was supported in part by the Goddard Space Flight Center, Greenbelt, MD, under Grant NAG 5-916.

K. Sayood is with the Department of Electrical Engineering, University of Nebraska, Lincoln, NE, 68588-0511.

J. C. Borkenhagen is with AT&T Bell Laboratories, Holmdel, NJ 07733.

IEEE Log Number 9144884.

difficult to analyze unless several simplifying assumptions are made about the quantizer input and the quantization noise. This is usually the strategy followed, with the quantizer being viewed as a source of additive white noise [20]. (It should be noted that this assumption is incorrect ([21]–[23]), however, its use simplifies the analysis which in turn provides useful insight into the workings of the overall system). Second, while the predictor design is based on a model for the source, this model might not be (and usually is not) matched to the source. This results in some residual correlation in the DPCM output, especially at low bit rates. However, most analyses which take into account the source model are based on the assumption that the model accurately represents the source.

In this section we study DPCM systems designed under the assumption that the source is an autoregressive process of unknown order. The predictor design is based on this assumption of an autoregressive source. However, as the order of the source process is unknown, the predictor may still be mismatched to the source. By mismatch we mean that the order of the process and hence the predictor coefficients are different from the autoregressive coefficients of the source.

First we will develop some results for the DPCM system without the quantizer in the feedback loop. The motivation for this is the same as the fine quantization assumption used by McDonald [25] and others; namely, simplicity. Without the quantizer in the loop, the DPCM encoder becomes a simple linear filter and consequently easy to analyze. Later, the quantizer will be introduced into the feedback loop, and using the results derived for the quantizerless system, the statistical structure of the encoder output will be studied.

The source is assumed to be an autoregressive process of order M, generated according to the difference equation

$$x(n) = \sum_{i=1}^{M} a_i x(n-i) + \epsilon(n) \tag{1}$$

where ϵ_n is a zero-mean white sequence with variance σ_ϵ^2. The following relationships can easily be obtained for the quantizerless system. If in the design of the DPCM system the source is assumed to be an autoregressive process of order N, the predictor output is given by

$$x(n \mid n-1) = \sum_{i=1}^{N} b_i x(n-i). \tag{2}$$

The prediction error $e(n)$ is given by

$$e(n) = x(n) - x(n \mid n-1). \tag{3}$$

If N is equal to M, and b_i is equal to a_i for all i, then, of course,

$$e(n) = \epsilon(n). \tag{4}$$

We have assumed our source to be an autoregressive process of order M. Such a process can be viewed as the output of a linear filter driven by white noise. Let $H_1(z)$ be the z-domain transfer function of a discrete time linear filter such that

$$H_1(z) = \frac{1}{1 - \sum_{i=1}^{M} a_i z^{-i}}. \tag{5}$$

Then, mixing operator and time domain notation,

$$x(n) = H_1 \epsilon(n). \tag{6}$$

Similarly, defining $H_2(z)$ as

$$H_2(z) = 1 - \sum_{i=1}^{N} b_i z^{-i}$$
$$e(n) = H_2 x(n). \tag{7}$$

Substituting the expression for $x(n)$ from (6)

$$e(n) = H_2 H_1 \epsilon(n) \tag{8}$$

where

$$H_2(z)H_1(z) = \frac{1 - \sum_{i=1}^{N} b_i z^{-i}}{1 - \sum_{i=1}^{M} a_i z^{-i}}. \tag{9}$$

Therefore,

$$e(n) = \sum_{i=1}^{M} a_i e(n-i) - \sum_{i=1}^{N} b_i \epsilon(n-i) + \epsilon(n) \tag{10}$$

and $e(n)$ is an ARMA (M,N) process. Notice that if $M = N$ and $a_i = b_i$ for all i, then

$$H_2(z)H_1(z) = 1 \tag{11}$$

and again

$$e(n) = \epsilon(n).$$

If the quantizer is now introduced into the loop, the prediction, and hence the prediction error, becomes contaminated by the quantization noise. For this situation, we obtain the following analogs of (2) and (3). The predictor output is given by

$$\hat{x}(n|n-1) = \sum_{i=1}^{M} b_i \hat{x}(n-i) \tag{12}$$

and the contaminated prediction error $\tilde{e}(n)$ is given by

$$\tilde{e}(n) = x(n) - \hat{x}(n \mid n-1). \tag{13}$$

Using an additive noise model for the quantizer, we can write the output of the quantizer $\hat{e}(n)$ as

$$\hat{e}(n) = \tilde{e}(n) + q(n) \tag{14}$$

where $q(n)$ is the quantization noise. The output of the DPCM receiver (and the input to the predictor) $\hat{x}(n)$ is given by

$$\hat{x}(n) = \hat{x}(n|n-1) + \hat{e}(n). \tag{15}$$

Substituting the expression for $\hat{e}(n)$ from (14)

$$\hat{x}(n) = \hat{x}(n|n-1) + \tilde{e}(n) + q(n). \tag{16}$$

Then using (13)

$$\hat{x}(n) = x(n) + q(n) \tag{17}$$

As a first step towards this objective, we use the definition of conditional probability to write L as

$$L = \frac{P\left[\theta_i = \alpha_j, \theta_{i-1} = \alpha_m, \hat{\theta}_i = \alpha_n\right]}{P\left[\theta_{i-1} = \alpha_m, \hat{\theta}_i = \alpha_n\right]}. \tag{25}$$

This can be rewritten as (26) shown below. In (26), $P[\theta_{i-1} = \alpha_m]$ appears both in the numerator and the denominator and can therefore be canceled. Also, assuming a memoryless channel, and using conditional independence,

$$P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_j, \theta_{i-1} = \alpha_m\right] = P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_j\right]. \tag{27}$$

L thus becomes

$$L = \frac{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_j\right] P[\theta_i = \alpha_j \mid \theta_{i-1} = \alpha_m]}{P\left[\hat{\theta}_i = \alpha_n \mid \theta_{i-1} = \alpha_m\right]}. \tag{28}$$

The numerator of L is now in terms of the transition probabilities of the source encoder and the channel transition probabilities. To obtain the denominator of L in the same form we make the following claim.

Claim:

$$P\left[\hat{\theta}_i = \alpha_n \mid \theta_{i-1} = \alpha_m\right] = \sum_l \left\{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l\right] P[\theta_i = a_l \mid \theta_{i-1} = \alpha_m]\right\}. \tag{29}$$

Proof of Claim: Let

$$D = \sum_l \left\{P\left[\hat{\theta}_i = a_n \mid \theta_i = \alpha_l\right] P[\theta_i = \alpha_l \mid \theta_{i-1} = \alpha_m]\right\} \tag{30}$$

$$= \sum_l \left\{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l\right] \frac{P[\theta_i = \alpha_l, \theta_{i-1} = \alpha_m]}{P[\theta_{i-1} = \alpha_m]}\right\} \tag{31}$$

$$= \frac{1}{P[\theta_{i-1} = \alpha_m]} \cdot \sum_l \left\{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l\right] P[\theta_{i-1} = \alpha_m, \theta_i = \alpha_l]\right\}. \tag{32}$$

Assuming a memoryless channel,

$$P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l\right] = P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l, \theta_{i-1} = \alpha_m\right]. \tag{33}$$

Therefore,

$$\begin{aligned} D &= \frac{1}{P[\theta_{i-1} = \alpha_m]} \sum_l \cdot \left\{P[\theta_{i-1} = \alpha_m, \theta_i = \alpha_l] P[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l, \theta_{i-1} = \alpha_m]\right\} \\ &= \frac{1}{P[\theta_{i-1} = \alpha_m]} \sum_l P\left[\hat{\theta}_i = a_n, \theta_{i-1} = \alpha_m, \theta_i = \alpha_l\right] \\ &= \frac{P\left[\hat{\theta}_i = \alpha_n, \theta_{i-1} = \alpha_m\right]}{P[\theta_{i-1} = \alpha_m]} \end{aligned} \tag{34}$$

which by the definition of conditional probability proves the claim.

Thus, the $L(j, m, n)$ metric can finally be written as

$$L = \frac{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_j\right] P[\theta_i = \alpha_j \mid \theta_{i-1} = \alpha_m]}{\sum_l \left\{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_l\right] P[\theta_i = \alpha_l \mid \theta_{i-1} = \alpha_m]\right\}} \tag{35}$$

(alternate derivations can be found in [27], [28]).

If we examine the expression for L we notice that it is made up of two sets of conditional probabilities

$$\begin{aligned} &P\left[\theta_i = \alpha_j \mid \hat{\theta}_i = \alpha_n\right], \qquad j, n = 1, \cdots, N \quad \text{and} \\ &P[\theta_i = \alpha_j \mid \theta_{i-1} = \alpha_m], \qquad j, m = 1, \cdots, N \end{aligned} \tag{36}$$

where N is the size of the source coder output alphabet. Thus, to obtain L we need to estimate $2N^2$ quantities. For most data compression systems this number is usually quite small. The estimation of these quantities was clearly not a large burden on the two-, three-, and four-bit DPCM systems we studied where $2N^2$ was 32, 128, and 512, respectively. Notice that the first set of conditional probabilities depends only on the channel statistics, and the second set of transition probabilities depends only on the source coder and source statistics. If we assume a binary symmetric model for our channel, then given the binary code used over the channel and an estimate of the channel probability of error, it is a simple task to obtain the first set of transition probabilities. To obtain the second set of transition probabilities, we found it most convenient to use a training sequence. The use of a training sequence raises the question of how robust this approach will be if the training sequence is not exactly similar to the test sequence. Our results in the next section show that the approach is reasonably robust in terms of differences between training and test sequences.

IV. Results

We simulated the proposed approach using differential PCM (DPCM) as our test system. It should be emphasized that

$$L = \frac{P\left[\hat{\theta}_i = \alpha_n \mid \theta_i = \alpha_j, \theta_{i-1} = \alpha_m\right] P[\theta_i = \alpha_j \mid \theta_{i-1} = \alpha_m] P[\theta_{i-1} = \alpha_m]}{P\left[\hat{\theta}_i = \alpha_n \mid \theta_{i-1} = \alpha_m\right] P[\theta_{i-1} = \alpha_m]}. \tag{26}$$

3.

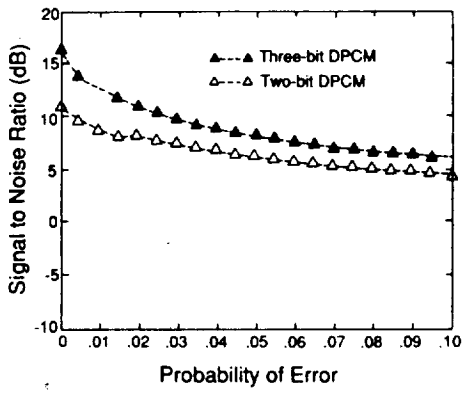


Fig. 7. Two- and three-bit proposed system performance in the presence of channel noise.

dictor coefficient is 0.778625. Notice the large improvement in the noisy channel performance over the previous figure. While the performance improvement with the JSCD is still on the order of 3 dB, the absolute value of the signal-to-noise ratio is such as to make the system feasible at relatively high probabilities of error. Figs. 5 and 6 repeat the results of the previous figure except the two-bit quantizer is replaced by three and four-bit quantizers, respectively. Notice that the performance improvements are in the range of 6–8 dB. The results are especially striking in the four-bit case where the use of the joint source/channel decoder improves the signal-to-noise ratio of the RDPCM system by about 8 dB at high probability of error. The improvement of the RDPCM system with the JSCD over the DPCM system is about 16 dB!

Because of the improvement in noisy channel performance obtained by using the Chang and Donaldson predictor, we have incorporated that into our design, and from now on when we mention the proposed system we mean a DPCM system with a 1 tap Chang and Donaldson predictor and the JSCD in front of the source decoder. The performance of the two- and three-bit proposed system is shown in Fig. 7. In some ways this is the most interesting of all the graphs. Notice that contrary to the classical DPCM system behavior shown in Fig. 1, the three-bit DPCM+JSCD consistently outperforms the two-bit system. Thus, the paradoxical situation of getting poorer performance for a higher rate no longer exists. Also the performance curves remain relatively flat, which is in marked contrast to the performance shown in Fig. 1. This indicates a more graceful degradation in performance as the channel becomes noisier.

While the objective performance of the proposed system (in terms of signal-to-noise ratio) is impressive, the final arbiter for any image coding scheme has to be the human eye. Figs. 8–10 present images coded using DPCM and the proposed system and transmitted over channels with different probability of error. Fig. 8 contains the test results for a two-bit system, while Figs. 9 and 10 contain results for three- and four-bit systems, respectively. In each case the image labeled "(a)" is DPCM coded with a channel probability of error of 0.02, while the image labeled "(b)" is coded with the proposed system under the same channel condition. The image labeled "(c)" is DPCM coded with a channel probability of

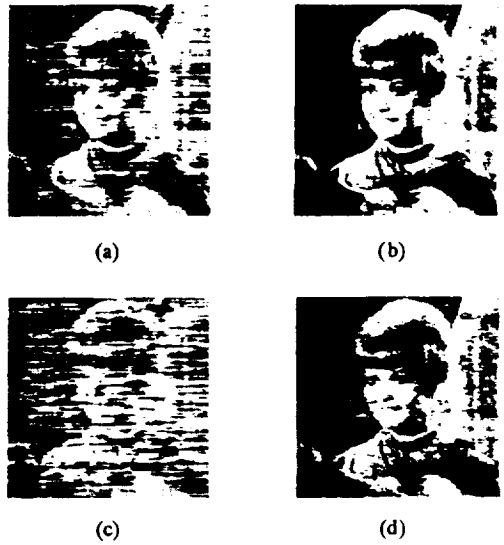


Fig. 8. U.S.C. GIRL image coded at two bits per pixel (a) using DPCM with channel error probability of 0.02, (b) using the proposed system with channel error probability of 0.02, (c) using DPCM with channel error probability of 0.1, (d) using the proposed system with channel error probability of 0.1.

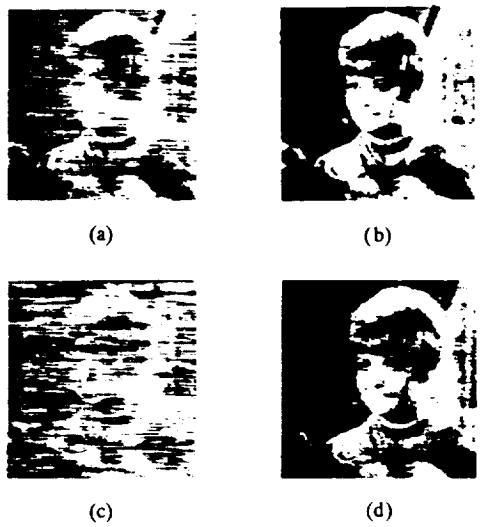


Fig. 9. U.S.C. GIRL image coded at three bits per pixel (a) using DPCM with channel error probability of 0.02, (b) using the proposed system with channel error probability of 0.02, (c) using DPCM with channel error probability of 0.1, (d) using the proposed system with channel error probability of 0.1.

error of 0.10, while the image labeled "(d)" is coded with the proposed system, with a channel probability of error of 0.10. The improvement in the perceptual quality of the images is truly striking, especially in the case of the channel with 0.10 probability of error. In this case the DPCM coded image is reduced to a nearly indistinguishable blur, while the image coded with the proposed system is reasonably clean. This is especially encouraging as it means that even for such high error rates the proposed system makes the channel usable for image transmission.

Finally, we examine the issue of performance indicators which could be used by the system designer to get an idea about the performance improvements available through the

this paper and the anonymous reviewers for their helpful suggestions.

REFERENCES

[1] J. L. Massey, "Joint source and channel coding," in *Communication Systems and Random Process Theory*, J. K. Skwirzynski, Ed. Amsterdam, The Netherlands: Sijthhoffi and Nordhoff, 1978, pp. 279–293.

[2] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

[3] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*. New York: McGraw-Hill, 1979.

[4] J. W. Modestino, D. G. Daut, and A. L. Vickers, "Combined source-channel coding of images using the block cosine transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1262–1274, Sept. 1981.

[5] D. Comstock and J. D. Gibson, "Hamming coding of DCT compressed images over noisy channels," *IEEE Trans. Commun.*, vol. COM-32, pp. 856–861, July 1984.

[6] K.-Y. Chang and R. W. Donaldson, "Analysis, optimization, and sensitivity study of differential PCM systems operating on noisy communication channels," *IEEE Trans. Commun.*, vol. COM-20, pp. 338–350, June 1972.

[7] R. Steele, D. J. Goodman, and C. A. McGonegal, "A difference detection and correction scheme for combatting DPCM transmission errors," *IEEE Trans. Commun.*, vol. COM-27, pp. 252–255, Jan. 1979.

[8] ——, "Partial correction of transmission errors in DPCM without recourse to error correction coding," *Electron. Lett.*, vol. 13, pp. 351–353, June 1977.

[9] R. C. Reininger, and J. D. Gibson, "Soft decision demodulation and transform coding of images," *IEEE Trans. Commun.*, vol. COM-31, pp. 572–577, Apr. 1983.

[10] R. J. Arguello, H. R. Sellner, and J. A. Stuller, "The effect of channel errors in the differential pulse-code-modulation transmission of sampled imagery," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 926–933, Dec. 1971.

[11] R. Steele, D. J. Goodman, and C. A. McGonegal, "DPCM with forced updating and partial correction of transmission errors," *Bell Syst. Tech. J.*, vol. 58, pp. 721–728, Mar. 1979.

[12] K. N. Ngan and R. Steele, "Enhancement of PCM and DPCM images corrupted by transmission errors," *IEEE Trans. Commun.*, vol. COM-30, pp. 257–264, Jan. 1982.

[13] D. J. Goodman and C.-E. Sundberg, "Combined source and channel coding for variable bit rate speech transmission," *Bell Syst. Tech. J.*, vol. 62, pp. 2017–2036, Sept. 1983.

[14] ——, "Transmission errors and forward error correction in embedded differential pulse code modulation," *Bell Syst. Tech. J.*, vol. 62, pp. 2735–2764, Nov. 1983.

[15] C. C. Moore and J. D. Gibson, "Self-orthogonal convolutional coding for the DPCM-AQB speech encoder," *IEEE Trans. Commun.*, vol. COM-32, Aug. 1984.

[16] A. J. Kurtenbach and P. A. Wintz, "Quantizing for noisy channels," *IEEE Trans. Commun. Technol.*, vol. COM-17, pp. 291–302, Apr. 1969.

[17] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: An approach to combined source–channel coding," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 827–838, Nov. 1987.

[18] E. Ayanoglu, and R. M. Gray, "The design of joint source and channel trellis waveform coders," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 855–865, Nov. 1987.

[19] K. Nitadori, "Statistical analysis of ΔPCM," *Electron. Commun. Japan*, vol. 48, Feb. 1965.

[20] J. B. O'Neal, Jr., "Signal to noise ratios for differential PCM," *IEEE Trans. Commun.*, vol. COM-19, pp. 568–570, Aug. 1971.

[21] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[22] R. C. Wood, "On optimum quantization," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 248–252, Mar. 1969.

[23] J. A. Bucklew and N. C. Gallagher, "A note on optimal quantization," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 365–366, May 1979.

[24] K. Sayood and J. D. Gibson, "Explicit additive noise models for uniform, and nonuniform MMSE quantization," *Signal Processing*, vol. 7, pp. 407–414, Dec. 1984.

[25] R. A. McDonald, "Signal-to-noise and idle channel performance of differential pulse code modulation systems—Particular application to voice signals," *Bell Syst. Tech. J.*, vol. 45, pp. 1123–1151, Sept. 1966.

[26] K. Sayood, "Analysis of differential PCM," in *Proc. Nineteenth Asilomar Conf. Circuit, Syst., and Comput.*, Nov. 1985, pp. 218–222.

[27] K. Sayood and J. C. Borkenhagen, "Utilization of correlation in low rate DPCM systems for channel error protection," in *Proc. IEEE Int. Conf. Commun.*, June 1986, pp. 1888–1892.

[28] J. C. Borkenhagen, "An application of maximum-likelihood estimation for error protection in data compression systems," M.S. thesis, Univ. Nebraska-Lincoln, Oct. 1986.

[29] S. Lin and D. J. Costello, Jr., *Error Control Coding*. Englewood Cliffs, NJ: Prentice-Hall, 1983.

[30] K. Sayood, and J. D. Gibson, "Maximum *a posteriori* joint source/channel coding," in *Proc. 22nd Annu. Conf. Inform. Sci. Syst.*, Princeton, NJ, Mar. 1988.

Khalid Sayood (S'78–M'82) was born in Pakistan in 1956. He received his undergraduate education from the Middle East Technical University, Ankara, Turkey. He received the B.S. and M.S. degrees from the University of Rochester, Rochester, NY, and the Ph.D. degree from Texas A&M University, College Station, TX, in 1977, 1979, and 1982, respectively, all in electrical engineering.

He joined the Department of Electrical Engineering at the University of Nebraska-Lincoln, in 1982, where he is currently serving as an Associate Professor. His current research interests include data compression, joint source/channel coding, communication networks, and biomedical applications.

Dr. Sayood is a member of Eta Kappa Nu and Sigma Xi.

Jay C. Borkenhagen (S'83–M'87) was born in Omaha, NE, on October 16, 1961. He received the B.S. and M.S. degrees in electrical engineering from the University of Nebraska-Lincoln, in 1984 and 1986, respectively.

Since 1986 he has been a Member of Technical Staff in the Data Network Analysis Department of AT&T Bell Laboratories, Holmdel, NJ, where he has been involved in the planning and design of large data networks.

Mr. Borkenhagen is a member of the IEEE Communications and Information Theory Societies and Eta Kappa Nu.

Appendix 2- Item 2

Presented at the 22nd Annual Conference on Information Science and Systems, March 1988, Princeton, New Jersey.

MAXIMUM APOSTERIORI JOINT SOURCE/CHANNEL CODING

Khalid Sayood*
Department of Electrical Engineering
University of Nebraska
Lincoln, Nebraska 68588-0511
(402) 472-6688

Jerry D. Gibson
Department of Electrical Engineering
Texas A&M University
College Station, TX 77843
(409)845-7441

1. INTRODUCTION

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be treated separately without any loss of performance for the overall system [1]. The basic design procedure is to select a source encoder which changes the source sequence into a series of independent, equally likely binary digits followed by a channel encoder which accepts binary digits and puts them into a form suitable for reliable transmission over the channel. However, the separation argument no longer holds if either of the following two situations occur:

i. The input to the source decoder is different from the output of the source encoder, which happens when the link between the source encoder and source decoder is no longer error free, or

ii. when the source encoder output contains redundancy.

Of course, case (i) occurs when the channel coder does not achieve zero error probability and case (ii) occurs when the source encoder is suboptimal. These two situations are common occurrences in practical systems where source or channel models are imperfectly known, complexity is a serious issue, or significant delay is not tolerable. Various approaches have been developed for such situations. They are usually grouped under the general heading of joint source/channel coding.

Most of the various joint source channel coding approaches can be classified in two main categories; (A) approaches which entail the modification of the source coder/decoder structure to reduce the effect of channel errors, and (B) approaches which examine the distribution of bits between the source and channel coders. The first set of approaches can be divided still further into two classes. One class of approaches examines the modification of the overall structure, while the other deals with the modification of the decoding procedure to take advantage of the redundancy in the output of the source coder.

To the first class belongs the work of Dunham & Gray [2] who proved the existence of joint source channel trellis coding systems for certain fidelity criteria, and a design of a joint source channel trellis coder presented by Ayanoğlu and Gray [3], where the design procedure is the generalized Lloyd algorithm. Further, Massey [4] and Ancheta [5] showed that for distortionless transmission of the source using linear joint source channel encoders, equivalent performance can be obtained with a significant reduction in complexity. Chang and Donaldson [6] propose modifications to the DPCM system to reduce the effect of channel errors, while Kurtenbach and Wintz [7] and Farvardin and Vaishampayan [8] study the problem of optimum quantizer design for noisy channels. Goodman and Sundberg [9,10] propose an embedded DPCM system which consists of a two bit DPCM and a two bit PCM system in parallel.

In the second class of category A, we include the work of Reininger and Gibson [11], who use the fact that coefficients in neighboring blocks in a transform coding scheme will not vary greatly, and thus use coefficients from neighboring blocks to correct a possible error, and the work of Steele, Goodman and McGonegal [12,13], who propose a difference detection and correction scheme for broadcast quality speech. In this scheme the receiver infers an error whenever an individual sample to sample difference is greater than the mean squared difference of a 21 sample sliding block. When an error is detected, the received sample is replaced by the output of a smoothing circuit. Ngan and Steele [14] use a similar method for recovering from errors in an image transmission system. Sayood and Borkenhagen [15,16] use the redundancy at the source coder output to perform sequence estimation.

*Supported by NASA Lewis Research Center [NAG 3-806] and NASA Goddard Space Flight Center [NAG 5-810]

The work of Modestino, Daut and Vickers [17] belongs to category B. In their study of transform coding they examine tradeoffs between allocating bits for source and channel coding. Comstock and Gibson [18] extend this work and provide an explicit mechanism for allocating bits between a source coder and a Hamming channel coder. Additionally, Moore and Gibson [19] study the allocation of bits between a DPCM coder and self orthogonal convolutional coding.

In this paper we present a maximum aposteriori probability (MAP) approach to joint source/channel coder design, which belongs to category A, and hence we explore a technique for designing joint source/channel coders, rather than ways of distributing bits between source coders and channel coders. We assume that the two nonideal situations referred to earlier are present. Our approach is as follows. For a nonideal source coder, we use MAP arguments to design a decoder which takes advantage of redundancy in the source coder output to perform error correction. Once the decoder is obtained, we analyze it with the purpose of obtaining "desirable properties" of the channel input sequence for improving overall system performance. We then propose an encoder design which incorporates these properties.

2. THE MAP DESIGN CRITERION

For a discrete memoryless channel (DMC), let the channel input alphabet be denoted by $A = \{a_0, a_1, \ldots, a_{M-1}\}$, and the channel input and output sequences by $Y = \{y_0, y_1, \ldots, y_{L-1}\}$ and $\hat{Y} = \{\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{L-1}\}$, respectively. If $\mathcal{A} = \{A_i\}$ is the set of sequences $A_i = \{a_{i,0}, a_{i,1}, \ldots, a_{i,L-1}\}$, $a_{i,k} \in A$, then the optimum receiver (in the sense of maximizing the probability of making a correct decision) maximizes $P[C]$, where

$$P[C] = \sum_{A_i} P[C|\hat{Y}]P[\hat{Y}].$$

This in turn implies that the optimum receiver maximizes $P[C|\hat{Y}]$. When the receiver selects the output to be A_k, then $P[C|\hat{Y}] = P[Y = A_k|\hat{Y}]$. Thus, the optimum receiver selects the sequence A_k such that

$$P[Y = A_k|\hat{Y}] \geq P[Y = A_i|\hat{Y}] \; \forall i.$$

Lemma 1

Let y_i be the input to a DMC. Given y_{i-1}, y_i is conditionally independent of $y_{n-k}, k > 1$. If $\hat{y}_0 = y_0$ then the optimum receiver selects a sequence A_i to maximize $\Pi_{i=1}^{L-1} p(y_i|y_{i-1}, \hat{y}_i)$.

Proof:

From the preceding result, the receiver tries to maximize $P(Y|\hat{Y})$. Using the chain rule we can write this as

$$\begin{aligned} P(Y|\hat{Y}) &= P(y_0, y_1, \ldots, y_{L-1}|\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{L-1}) \\ &= P(y_{L-1}|y_{L-2}, y_{L-3}, \ldots, y_0, \hat{y}_0, \ldots, \hat{y}_{L-1}) \\ &\quad P(y_{L-2}|y_{L-3}, \ldots, y_0, \hat{y}_0, \ldots, \hat{y}_{L-1}) \ldots P(y_0|\hat{y}_0, \ldots, \hat{y}_{L-1}) \end{aligned}$$

The last factor on the right hand side (RHS) is equal to one. Using the assumption of the DMC, we obtain

$$P(Y|\hat{Y}) = \Pi_{i=1}^{L-1} p(y_i|y_{i-1}, \hat{y}_i). \qquad (1)$$

□

The lemma addresses the situation in case (ii), i.e., the situation in which the source coder output (which is also the channel input sequence) contains redundancy. Using this lemma, we can design a decoder which will take advantage of dependence in the channel input sequence.

3. DECODER DESIGN

The lemma of the previous section provides the mathematical structure for the decoder. The physical structure can be easily obtained by examining the quantity to be maximized. The decoder maximizes $P(Y|\hat{Y})$ or equivalently $\log P(Y|\hat{Y})$, but

$$\log P(Y|\hat{Y}) = \sum \log P(y_i|\hat{y}_i, y_{i-1}) \tag{2}$$

and various solutions exist for the maximization of additive path metrics. To implement this decoder we need to be able to compute the path metric. This task is considerably eased by the following lemma.

Lemma 2 Let A be the channel input alphabet and $\{y_i\}$ and $\{\hat{y}_i\}$ be the input and output sequences of a DMC. Then

$$P[y_i = a_j | y_{i-1} = a_m, \hat{y}_i = a_n] =$$

$$\frac{P[\hat{y}_i = a_n | y_i = a_j] P[y_i = a_j | y_{i-1} = a_m]}{\sum_l P[y_i = a_l | y_{i-1} = a_m] P[\hat{y}_i = a_n | y_i = a_l]} \tag{3}$$

Proof: See [16].

The expression on the RHS of (3), while it looks more complicated, is actually a much more tractable form of the desired conditional probability. Note that this expression is a function of two distinct sets of transition probabilities, the channel transition probabilities and the source coder output transition probabilities. As the channel transition probabilities depend only on the channel, and the source coder output transition probabilities depend only on the source coder and source probabilities, the two sets of transition probabilities can be estimated independently. The two can then be combined according to (3) to construct a M x M x M lookup table for use in decoding. If the source coder or source changes, the only parameters to be modified are the source coder output transition probabilities. Results using a DPCM source coder with an image as the source are presented in Section 6.

4. DECODER ANALYSIS

In the previous section we developed a scheme for providing error correction using the redundancy in the channel input sequence, or the source coder output. We looked at the design of the decoder given a source coder or channel input sequence with some rather general statistical properties. In this section we examine the reverse problem. That is, given the decoder obtained in the previous section we look for "desired properties" of the channel input sequence and, hence, the source coder.

To obtain the desired properties we need to examine the factors involved in the error correcting capability of the decoder. Toward this end let us examine the following situation. Referring to Figure 1, assume that the correct sequence of transmitted codewords is $a_0a_0a_0$. An error occurs if the path metric for $a_0a_ja_0$ is greater than the path metric of $a_0a_0a_0$. Assume $\hat{y}_1 = a_n$ and $\hat{y}_2 = a_0$. An error occurs if the following quantity is positive.

$$\log \frac{P[\hat{y}_1 = a_n | y_1 = a_j] P[y_1 = a_j | y_0 = a_0]}{\sum_l P[y_1 = a_l | y_0 = a_0] P[\hat{y}_1 = a_n | y_1 = a_l]}$$

$$+ \log \frac{P[\hat{y}_2 = a_0 | y_2 = a_0] P[y_2 = a_0 | y_1 = a_j]}{\sum_l P[y_2 = a_l | y_1 = a_j] P[\hat{y}_2 = a_0 | y_2 = a_l]}$$

$$- \log \frac{P[\hat{y}_1 = a_n | y_1 = a_0] P[y_1 = a_0 | y_0 = a_0]}{\sum_l P[y_1 = a_l | y_0 = a_0] P[\hat{y}_1 = a_n | y_1 = a_l]}$$

$$- \log \frac{P[\hat{y}_2 = a_0 | y_2 = a_0] P[y_2 = a_0 | y_1 = a_0]}{\sum_l P[y_2 = a_l | y_1 = a_0] P[\hat{y}_2 = a_0 | y_2 = a_l]} \tag{4}$$

Defining

$$d_{ij} = \text{Hamming distance between } a_i \text{ and } a_j$$

$$g_{lk} = P[y_i = a_l | y_{i-1} = a_k] \tag{5}$$

then using (5) and simplifying (4), an error occurs if

$$(d_{nj} - d_{n0}) \log\left(\frac{p}{1-p}\right) + \log \frac{g_{j0}}{g_{00}} + \log \frac{g_{0j}}{g_{00}} - \log \frac{\sum_l \left(\frac{p}{1-p}\right)^{d_{nl}} g_{lj}}{\sum_l \left(\frac{p}{1-p}\right)^{d_{nl}} g_{l0}} > 0 \tag{6}$$

Defining $\alpha = \frac{1-p}{p}$, (6) can be rewritten as

$$(d_{n0} - d_{nj}) \log \alpha + \log \frac{g_{j0}}{g_{00}} + \log \frac{g_{0j}}{g_{00}} - \log \frac{\sum_l (\alpha^{-d_{nl}} g_{lj})}{\sum_l (\alpha^{-d_{nl}} g_{l0})} > 0 \tag{7}$$

or

$$d_{n0} - d_{nj} > \frac{1}{\log \alpha} \left(\log \frac{g_{00}}{g_{j0}} + \log \frac{g_{00}}{g_{0j}} + \log \frac{\sum_l \alpha^{-d_{nl}} g_{lj}}{\sum_l \alpha^{-d_{nl}} g_{l0}} \right) \tag{8}$$

The left hand side is maximized when $j = n (d_{nj} = 0)$. Thus an error occurs when the number of bit errors, which in this case is d_{n0}, is greater than the quantity on the LHS of (8) or

$$d_{n0} > \frac{1}{\log \alpha} \left(\log \frac{g_{00}}{g_{j0}} + \log \frac{g_{00}}{g_{0j}} + \log \frac{\sum_l \alpha^{-d_{nl}} g_{lj}}{\sum_l \alpha^{-d_{nl}} g_{l0}} \right). \tag{9}$$

The alternative path shown in Figure 1 is only one of possible paths. Another longer alternative path is shown in Figure 2.
In this case the number of errors required to take the alternative path is given by

$$d_{n0} > d_{nm} + \frac{1}{\log \alpha} \left(\log \frac{g_{00}}{g_{lmn}} + \log \frac{g_{00}}{g_{0m}} \right)$$

$$+ \frac{1}{\log \alpha} \left(\log \frac{\sum \alpha^{-d_{jl}} g_{l0}}{\sum \alpha^{-d_{jl}} g_{lm}} + \log \frac{\sum \alpha^{-d_{nl}} g_{l0}}{\sum \alpha^{-d_{nl}} g_{lm}} \right) \tag{10}$$

Notice that the number of errors required to take a longer incorrect path (a path with more branches) is larger than for a shorter incorrect path. To make our statements more concrete, we define a parameter we call the error correction capability I as

$$I = 1 - H(Y_n | Y_{n-1}) / \log M =$$

$$1 - \frac{1}{\log M} \sum_{l,k} p(y_n = a_l, y_{n-1} = a_k) \log \frac{1}{P(y_n = a_l | y_{n-1} = a_k)}$$

$$= 1 - \frac{1}{\log M} \sum_{l,k} p(y_n = a_l, y_{n-1} = a_k) \log \frac{1}{g_{lk}} \tag{11}$$

where M is the size of the channel input alphabet. We immediately note the following properties of I

(i) I is a convex cup function of the conditional probabilities $\{p(y_n|y_{n-1})\}$.
(ii) $0 \le I \le 1$.

Further properties of I are developed in the following lemmas.

Lemma 3: If I is zero for a particular channel input sequence, the decoder will not correct any errors.

Proof:

I is zero when

$$\sum p(y_n, y_{n-1}) \log \frac{1}{g_{lk}} = \log M$$

This is true when $g_{lk} = \log \frac{1}{M}$ for all l, k. In this condition the right hand side of (9) is zero giving the desired result.

□

Lemma 4: If I is one for a particular input sequence, the decoder obtains the correct sequence with probability one.

Proof:

For I to be one, $H(y_n|y_{n-1})$ has to be zero. This is true if for each k_0 there exists an l_0 such that

$$g_{lk_0} = \begin{cases} 1, & l = l_0 \\ 0, & l \neq l_0. \end{cases}$$

This in turn implies that there exists some i_0 such that

$$p(A_i) = \begin{cases} 1, & i = i_0 \\ 0, & i \neq i_0. \end{cases}$$

Thus

$$P(Y = A_i | \hat{Y}) = \begin{cases} 1, & i = i_0 \\ 0, & i \neq i_0 \end{cases}$$

and the decoder will pick the correct sequence with probability one. □

The above two lemmas provide a relationship between the value of I and the error correcting capability of the decoder, for the extreme values of I. To obtain an insight into the relationship for other values of I we look at a simplified version of (9). Assume that the size of the channel input alphabet is two, then (9) simplifies to

$$d_{10} > \frac{1}{\log \alpha}\left(\log\frac{g_{00}}{g_{10}} + \log\frac{g_{00}}{g_{01}} + \log\left(\frac{g_{01}+\alpha^{-d_{01}}g_{11}}{g_{00}+\alpha^{-d_{01}}g_{10}}\right)\right). \quad (12)$$

Noting that $g_{00} = 1-g_{10}, g_{01} = 1-g_{11}$, and for the right hand side to be positive $g_{00} > \frac{1}{2}$ and $g_{01} < \frac{1}{2}$ we can rewrite (12) as

$$d_{10} > \frac{1}{\log \alpha}\left(\log\frac{g_{00}}{g_{10}} + \log\left(\frac{1+\alpha^{-d_{01}}\frac{g_{11}}{g_{01}}}{1+\alpha^{-d_{01}}\frac{g_{10}}{g_{00}}}\right)\right). \quad (13)$$

In (13) the larger the right hand side the greater is the error correcting capability of the receiver. The right hand side can be increased by decreasing g_{01} below $\frac{1}{2}$. Thus the error correcting capability increases as g_{01} decreases below $\frac{1}{2}$. If we examine I we find that I increases as g_{01} decreases below $\frac{1}{2}$. This is because

$$H(Y_n|Y_{n-1}) = p_0\left((g_{00})\log\frac{1}{g_{00}} + g_{10}\log\frac{1}{g_{10}}\right) + p_1\left(g_{01}\log\frac{1}{g_{01}} + p_1(1-g_{01})\log\frac{1}{1-g_{01}}\right) \quad (14)$$

decreases with g_{01} decreasing below $\frac{1}{2}$. Thus for this simple example, an increase in I means an increase in the error correcting capability of the decoder.

5. ENCODER DESIGN

In the previous section we obtained desirable properties for the channel input/encoder output sequence. In this section we examine ways of incorporating these desireable properties into the encoder. We wish to do this without decreasing the redundancy removal capability of the source coder, and if possible, without increasing the transmitted bit rate. To see how to approach this problem let us first examine the source coder for noiseless channels in some detail.

In general, a source coder consists of two operations, data compression and data compaction [20]. The data compression operation usually consists of redundancy removal and involves some loss of information. Examples of data compression schemes are DPCM, transform coding and vector quantization. The data compaction schemes are information preserving. They may result in a variable rate out. Examples include Huffman coding and runlength coding. Generally in discussions of joint source/channel coder design, the data compaction operations are not included. The reason for this is that due to the variable rate output, the data compaction schemes are highly vulnerable to channel noise and, therefore, are not considered for noisy channel applications.

A possible way of achieving our objectives is to insert another operation between the data compression and data compaction steps as shown in Figure 3.

To satisfy our objectives, the Π operator should have the following properties.

(a) The Π operator should perform distortionless encoding.

(b) The Π operator should increase the error correcting capability.

(c) The Π operator should not increase the bit rate. For the case where the data compaction scheme is a Huffman coder, this is equivalent to the condition that the output entropy not be greater than the input entropy.

An example of the Π operator which satisfies (a) and (b) and which can be modified to satisfy (c) functions as follows. Let the input to Π be selected from the alphabet

$$A = \{a_0, a_1, \ldots, a_{N-1}\},$$

and let the output alphabet be denoted by

$$S = \{s_0, s_1, \ldots, s_{N^2-1}\}.$$

Then the input/output mapping is given by

$$x_n = a_i, x_{n-1} = a_j \Longrightarrow y_n = s_{jN+i}. \quad (15)$$

The effect of the Π operator is to increase the distance between alternative sequences. To see this, let us construct a simple example. Let $A = \{a_0, a_1\}$ and $S = \{s_0, s_1, s_2, s_3\}$ then

$$\begin{aligned} y_n = s_0 \quad &\text{if} \quad x_n = a_0 \quad \text{and} \quad x_{n-1} = a_0, \\ y_n = s_1 \quad &\text{if} \quad x_n = a_1 \quad \text{and} \quad x_{n-1} = a_0, \\ y_n = s_2 \quad &\text{if} \quad x_n = a_0 \quad \text{and} \quad x_{n-1} = a_1, \\ y_n = s_3 \quad &\text{if} \quad x_n = a_1 \quad \text{and} \quad x_{n-1} = a_1. \end{aligned}$$

In this case if $y_n = s_0$, y_{n+1} cannot be s_2 or s_3 because $y_n = s_0$ means $x_n = a_0$, and $y_{n+1} = s_2$ or s_3 means $x_n = a_1$. Thus a decoded sequence cannot have s_2 or s_3 following s_0.

For simplicity let us ignore the Huffman coder and assign fixed length codewords to the s_i as

$$s_0 : 00,\ s_1 : 01,\ s_2 : 10,\ s_3 : 11$$

Now suppose the transmitted sequence was the all zero sequence, the metric used was the Hamming distance, and the received sequence is 00001000000000; that is, there is an error in the fifth bit. If the receiver decoded the first four bits as s_0s_0 then it cannot decode the fifth and sixth bits as s_2 for the reason noted above. The only two options are decoding them as s_0 or s_1. If we decoded them as s_0, we could continue decoding the rest of the sequence as $s_0s_0\ldots$, and the Hamming distance between the received and decoded sequence would be one. If we decoded them as s_1, we would have to decode the next set of two bits as s_2 or s_3 because s_0 cannot follow s_1. Decoding as s_2 gives the smallest Hamming distance so we decode the seventh and eighth bit as s_2. This gives a total Hamming distance of two for the incorrect path. Thus the receiver will select the correct path (the path with the smallest Hamming distance).

6. SIMULATION RESULTS

We present the results of simulating two different systems in this section. The first set of results were obtained using a nonideal source coder with the decoder proposed in Section 3. The second set of results pertain to the system proposed in the previous section. In both cases the data compression scheme is a DPCM system with a fixed one tap predictor and a nonuniform Lloyd-Max quantizer.

The source for the first set of results is the USC:GIRL image. The source coder output transition probabilities were obtained using a training set. The training image was the USC COUPLE image. The performance measure was the Peak-signal-to-noise-ratio (PSNR) defined as

$$PSNR(dB) = 10\log_{10}\left(\frac{\sum(255)^2}{\sum(x_i-\hat{x}_i)^2}\right)$$

where x_i is the input to the source coder while $\hat{x}_i$ is the output of the source decoder. Figure 4 shows the performance comparison for a two bit per pixel system. Most of the performance improvement is available at high probabilities of error. At these probabilities of error, however, the improvement is substantial. Figure 5 shows the same kind of results for a four bit per pixel system. The performance improvement for this case are even more substantial than those for the two-bit system. Two things are especially noteworthy in these results. The first one is that the performance improvement does not really become significant until the channel is very noisy. The other is that the performance curve in the high noise region is relatively flat. This means that even very noisy channels may be usable for image transmission. Further results including perceptual results can be found in [16].

The second set of results were obtained using the approach proposed in Section 5. The source encoder was replaced by the proposed joint source/channel coder. The Π operator used is the one described in the previous section. The source again was the USC:GIRL image, and end of line synchronization was assumed. The performance comparison is shown in Figure 6. Note that unlike the previous case, the performance improvement occurs at both low and high error probabilities. This makes the scheme especially useful for transmission at low error rates.

7. CONCLUSIONS

In this paper we have presented a MAP approach to joint source/channel coder design. The approach is based in part on the fact that source coders are, in general, nonideal and, therefore, cannot remove all redundancy from a source. This nonidenlity is taken advantage of, by a MAP decoder, to correct errors. The decoder is analyzed to obtain desired properties for the encoder output sequence. A joint source/channel encoder design approach is presented which incorporates the desired properties, and examples are given which show that considerable performance improvements can be obtained with the proposed approach.

8. REFERENCES

1. C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

2. J. G. Dunham and R. M. Gray, "Joint Source and Channel Trellis Encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 516–519, July 1981.

3. E. Ayanoğlu and R. M. Gray, "The Design of Joint Source and Channel Trellis Waveform Coders," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 855–865, November 1987.

4. J. L. Massey, "Joint Source and Channel Coding," in *Communication Systems and Random Process Theory*, J. K. Skwirzynski, ed., Sijthoff and Nordhoff: Netherlands, pp. 279–293, 1978.

5. T. C. Anchela, Jr., "Bounds and Techniques for Linear Source Coding," Ph.D. dissertation, Dept. of Electrical Engineering, Univ. of Notre Dame, August 1977.

6. K-Y Chang and R. W. Donaldson, "Analysis, Optimization, and Sensitivity Study of Differential PCM Systems Operating on Noisy Communication Channels," *IEEE Trans. Commun.*, vol. COM-20, pp. 338–350, June 1972.

7. A. J. Kurtenbach and P. A. Wintz, "Quantizing for Noisy Channels," *IEEE Trans. Commun. Technol.*, vol. COM-17, pp. 291–302, April 1969.

8. N. Farvardin and V. Vaishampayan, "Optimal Quantizer Design for Noisy Channels: An approach to Combined Source-Channel Coding," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 827–838, November 1987.

9. D. J. Goodman and C. E. Sundberg, "Combined Source and Channel Coding for Variable Bit Rate Speech Transmission," *Bell Syst. Tech. J.*, vol. 62, pp. 2017–2036, September 1983.

10. D. J. Goodman and C. E. Sundberg, "Transmission Errors and Forward Error Correction in Embedded Differential Pulse Code Modulation," *Bell Syst. Tech. J.*, vol. 62, pp. 2735–2764, November 1983.

11. R. C. Reininger and J. D. Gibson, "Soft Decision Demodulation and Transform Coding of Images," *IEEE Trans. Commun.*, vol. COM-31, pp. 572–577, April 1983.

12. R. Steele, D. J. Goodman, and C. A. McGonegal, "A Difference Detection and Correction Scheme for Combatting DPCM Transmission Errors," *IEEE Trans. Commun.*, vol. COM-27, pp. 252–255, January 1979.

13. R. Steele, D. J. Goodman, and C. A. McGonegal, "Partial Correction of Transmission Errors in DPCM without recourse to Error Correction Coding," *Elec. Lett.*, vol. 13, pp. 351–353, June 1977.

14. K. N. Ngan and R. Steele, "Enhancement of PCM and DPCM Images Corrupted by Transmission Errors," *IEEE Trans. Commun.*, vol. COM-30, pp. 257–269, January 1982.

15. K. Sayood and J. C. Borkenhagen, "Utilization of Correlation in Low Rate DPCM Systems for Channel Error Protection," *Proceedings IEEE International Conference on Communications*, June 1986, pp. 1888–1892.

16. K. Sayood and J. C. Borkenhagen, "Use of Residual Redundancy in the Design of Joint Source/Channel Coders," submitted to *IEEE Transactions on Communications*.

17. J. W. Modestino, D. G. Daut, and A. L. Vickers, "Combined Source-Channel Coding of Images Using the Block Cosine Transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1262–1274, September 1981.

18. D. Comstock and J. D. Gibson, "Hamming Coding of DCT Compressed Images Over Noisy Channels," *IEEE Trans. Commun.*, vol. COM-32, pp. 856–861, July 1984.

19. C. C. Moore and J. D. Gibson, "Self-Orthogonal Convolutional Coding for the DPCM-AQB Speech Encoder," *IEEE Trans. Commun.*, vol. COM-32, August 1984.

20. R. E. Blahut, *Principles and Practice of Information Theory*, Addison-Wesley, 1987.

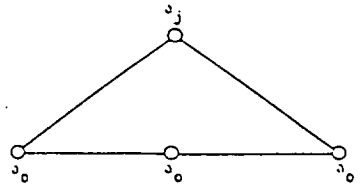


Figure 1. Alternative Paths at the Receiver

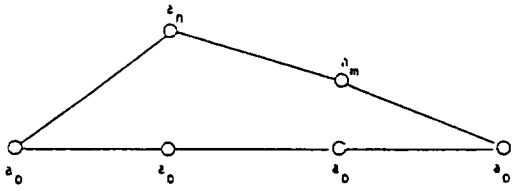


Figure 2. Longer alternative paths

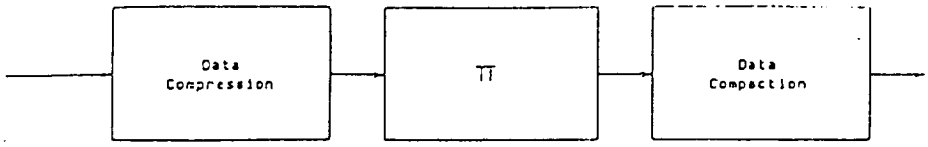


Figure 3. Proposed Joint Source/Channel Coder

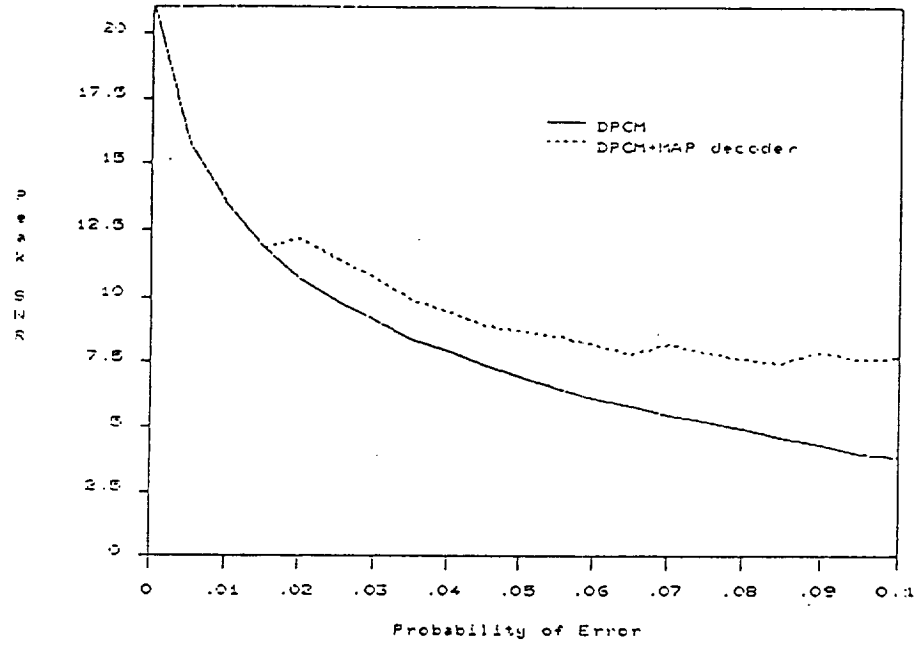


Figure 4. Performance of two-bit DPCM with MAP decoder

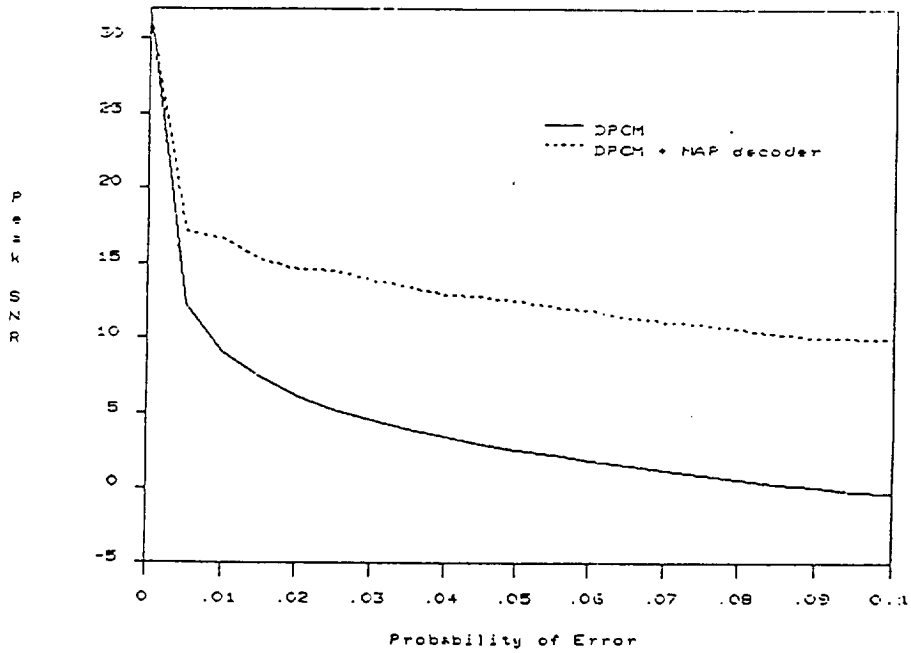


Figure 5. Performance of four-bit DPCM with MAP decoder

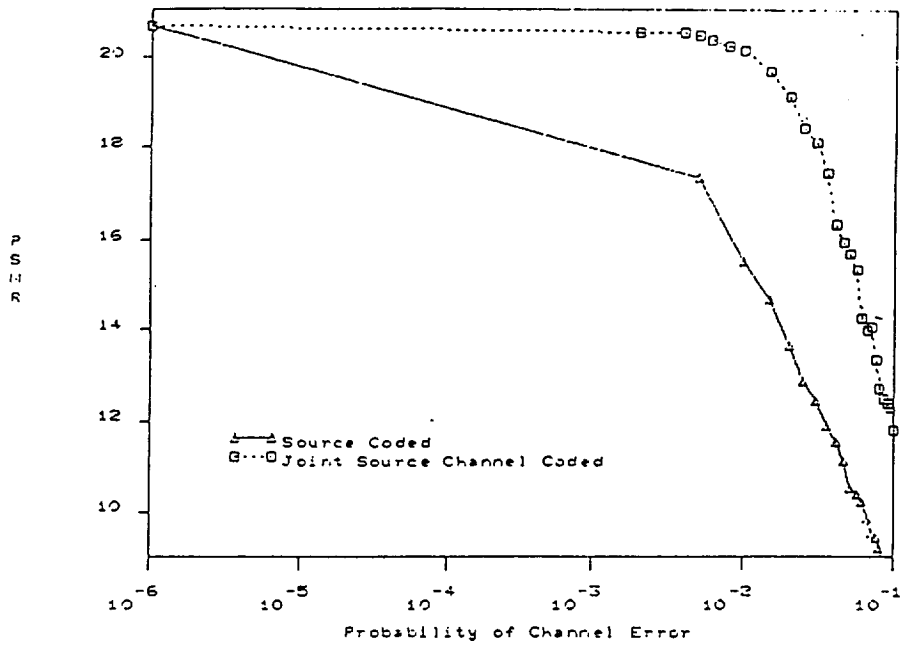


Figure 6. Performance Results of Joint Source/Channel Coding Scheme

Appendix 2- Item 3

IMPLEMENTATION ISSUES IN MAP JOINT SOURCE/CHANNEL CODING

Khalid Sayood* , Jerry D. Gibson@ and Fuling Liu*

*Department of Electrical Engineering
University of Nebraska
Lincoln, NE 68588-0511

@Department of Electrical Engineering
Texas A&M University
College Station, TX 77843

ABSTRACT

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be implemented separately without any loss of optimality. However, the assumption underlying this result may at times be violated in practice. Various joint source/channel coding approaches have been developed for handling such situations. A MAP approach to joint source/channel coding has been proposed which uses a MAP decoder and a modification of the source coder to provide error correction. We present various implementation strategies for this approach and provide results for an image coding application.

I. Introduction

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be treated separately without any loss of performance as compared to an optimum system [1]. The basic design procedure implied by Shannon's theorems consists of designing a source encoder which changes the source sequence into a series of (approximately) independent, equally likely binary digits followed by a channel encoder which accepts binary digits and puts them into a form suitable for reliable transmission over the channel 9 [2]. One aspect of the overall optimum system not addressed by Shannon is any increase in system complexity that results from this separation, and Massey [3] and Ancheta [4] showed that for distortionless transmission of the source under the constraint of linear source and channel coders, a significant reduction in complexity with equivalent performance can be achieved by using a linear joint source/channel coder. Their scheme also differs from most data compression systems in that the bulk of the system complexity is transferred to the receiver.

The theorem that provides the justification for the separate design of the source coder and the channel coder, often called the Information Transmission Theorem [2], assumes that both the source encoder/decoder pair and the channel encoder/decoder pair are operating in an optimal fashion. Specifically, the source encoder is assumed to present the channel encoder with a sequence suitable for optimal channel coding, and the channel encoder/decoder pair is assumed to reproduce the source encoder output at the source decoder input with negligible distortion. Unfortunately, there are practical situations where these assumptions are violated--namely, when the source encoder output contains redundancy, which occurs if the source encoder is suboptimal, and when the source decoder input differs from the source encoder output, which is a result of channel errors. These two situations are common occurrences in practical communication systems where source and/or channel models are imperfectly known, complexity is a serious issue, or significant delay is not tolerable. Various approaches have been developed to handle these two situations. These include approaches in which the source and channel coding operations are truly integrated [3-6], approaches that cascade known source coders with known channel coders and allocate the fixed bit rate to the source coder and channel coder to maximize system performance [7-15], approaches in which the source coder and/or receiver is modified to account for the presence of a given noisy channel [16-26], and approaches which use some knowledge of the source and source coder properties to detect channel errors and compensate for their effects [27-35]. The research described in this paper is concerned with the implementation of a joint source/channel coder design which was an extension of the work presented in [31,32]. This approach utilizes structure in the source encoder output by using a MAP decoder to correct errors introduced by the channel.

*Supported by NASA Lewis Research Center (NAG-3-806) and NASA Goddard Space Flight Center (NAG 5-916).

II. Previous Work

Based on the MAP design criteria, a decoder structure was proposed in [32] which takes advantage of redundancy in the channel input sequence to provide error correction. The decoder maximizes the quantity $\log P(Y \mid \hat{Y})$ where

$$Y = (y_1, y_2, \ldots, y_L)$$

is the channel input sequence while

$$\hat{Y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_L)$$

is the channel output sequence. If a Markov model is imposed on the channel input sequence, the path metric can be written as

$$\log P(Y \mid \hat{Y}) = \Sigma \log P(y_i \mid \hat{y}_i, y_{i-1}) \tag{1}$$

and

$$P(y_i = a_j \mid y_{i-1} = a_m, \hat{y}_i = a_n) \tag{2}$$

$$= \frac{P(\hat{y}_i = a_n \mid y_i = a_j) P(y_i = a_j \mid y_{i-1} = a_m)}{\Sigma_l P(\hat{y}_i = a_n \mid y_i = a_l) P(y_i = a_l \mid y_{i-1} = a_m)}$$

The proof of the above can be found in [31,32].

Based on analysis of the decoder a parameter called the error correction capability was defined in [35] as

$$I = 1 - H(y_n \mid y_{n-1})/\log M \tag{3}$$

We noted that a desirable property of a joint source channel coder would be to increase I. The approach proposed for this requires the modification of the source coder. In general, a source coder consists of two operations, data compression and data compaction [36]. The data compression operation usually consists of redundancy removal and involves some loss of information. Examples of data compression schemes are DPCM, transform coding and vector quantization. The data compaction schemes are information preserving. They may result in a variable rate out. Examples include Huffman coding and runlength coding. Generally in

discussions of joint source/channel coder design, the data compaction operations are not included. The reason for this is that due to the variable rate output, the data compaction schemes are highly vulnerable to channel noise and, therefore, are not considered for noisy channel applications.

A possible approach to achieving the objective of increasing I is to insert an invertible transoperation between the data compression and data compaction stages. An example of such an operation called the π operator, was presented in [35]. The operation can be described as follows. Let the input to the π operator be selected from the alphabet

$$S = (s_0, s_1, s_2,\ldots,s_{N-1}),$$

and let the output alphabet be denoted by

$$A = (a_0, a_1, a_2,\ldots,a_{N^2-1}).$$

Then the input/output mapping is given by

$$x_n = s_i, x_{n-1} = s_j \rightarrow y_n = a_{jN+i}.$$

This operator and its effects are described in more detail in [35].

While this approach achieves the objective of increasing the error correcting capability, it also results in a variable rate system. For this situation the branch metric of the form of (2) becomes difficult to implement. We explain these difficulties and propose implementatable approximations of the metrics in the next section. Section V contains simulation results which demonstrate the viability of these approximations. The use of a variable rate coder also complicates the structure of the decoder. In Section IV we present a modified Viterbi decoder which can be used with variable rate coder.

III. Development of the Path Metric

Before we begin our discussion of the path metric for variable rate case we need to summarize the derivation of (2). The derivation consists of two steps. First we show that

$$P(\hat{y}_i = a_j \mid y_{i-1} = a_m, \hat{y}_i = a_n) \tag{4}$$
$$= \frac{P(\hat{y}_i = a_n \mid y_i = a_j)P(y_i \mid y_{i-1})}{P(y_i = a_m \mid y_{i-1} = a_m)}$$

Then we show that the denominator can be written as

$$P(\hat{y}_i = a_n \mid y_{i-1} = a_m) = \Sigma_l P(\hat{y}_i = a_n \mid y_i \tag{5}$$
$$= a_l)P(y_i = a_l \mid y_{i-1} = a_m)$$

Note first that in this derivation the channel input alphabet and output alphabet are the same. We have assumed hard decision at the output of the channel and for a fixed rate coder this translates into identical alphabets at the input and output of the channel. For the case where we have variable rate codes there is a subtle difference. In fact, there are two different ways in which we can view the output of the channel. The first approach is to assume that there is a Huffman decoder at the output of the channel. The Huffman decoder output alphabet is the same as the joint source/channel (JSC) coder output alphabet. Thus the branch metric as derived in (2) can be used directly. However, now the computation of the individual factors of the branch metric becomes somewhat more involved. Specifically, consider the calculation of $P(\hat{y}_i = a_n \mid y_i = a_j)$, where the channel is assumed to be a binary symmetric channel with known crossover probability p. Let $l(a_i)$ be the number of bits in the binary codeword corresponding to the symbol a_i.

If $l(a_n) = l(a_j)$, as is the case when a fixed rate code is used, then

$$P(\hat{y}_i = a_n \mid y_i = a_j) = p^{d_{nj}}(1 - p)^{n-d_{nj}} \tag{6}$$

where d_{nj} is the Hamming distance between the binary codewords corresponding to a_n and a_j and n is the number of bits in each of the codewords. However, when $l(a_n) \neq l(a_j)$, the calculation may not be as simple. To see this we need to introduce some more notation. Let the codeword corresponding to a_n be represented by

$$a_n = (a_{n_1}, a_{n_2},\ldots,a_{n_{l(a_n)}})$$

Then, if $l(a_n)$ is less than $l(a_j)$

$$P(\hat{y}_i = a_n \mid y_i = a_j) = \Pi_{k=1}^{l(a_n)} Pr(a_{n_k} \mid a_{j_k}) \tag{7}$$

and the calculation is still relatively straightforward. However, if $l(a_n)$ is greater than $l(a_j)$,

$$P(\hat{y}_i = a_n \mid y_i = a_j) = \Sigma_l P(\hat{y}_i = a_n, y_{i+1} = a_l \mid y_i = a_j) \tag{8}$$

or in more familiar terms

$$P(\hat{y}_i = a_n \mid y_i = a_j) = \Sigma_l P(\hat{y}_i = a_n \mid y_i \tag{9}$$
$$= a_j, y_{i+1} = a_l)P(y_{i+1} = a_l \mid y_i = a_n)$$

where we have used the chain rule and the Markov property of JSC coder output. The second factor in the summand is simply the transition probability of the JSC coder output while the first factor can be calculated as

$$P(\hat{y}_i = a_n \mid y_i = a_j, y_{i+1} = a_l)$$
$$= \Pi_{k=1}^{l(a_j)} Pr(a_{n_k} \mid a_{j_k}) \Pi_{k=l(a_j)+1}^{l(a_n)} Pr(a_{n_k} \mid a_{l_{k-l(a_j)}})$$

as long as $l(a_n)$ is less than or equal to $l(a_j) + l(a_l)$. If not we simply repeat the process again to obtain

$$P(\hat{y}_i = a_n \mid y_i = a_j, y_{i+1} = a_l) = \Sigma_h P(\hat{y}_i = a_n, y_{i+2} = a_h \mid y_i \tag{10}$$
$$= a_j, y_{i+1} = a_l) = \Sigma_h P(\hat{y}_i = a_n \mid y_i = a_j, y_{i+1} = a_l, y_{i+2}$$
$$= a_h)P(y_{i+2} = a_h \mid y_{i+1} = a_l)$$

Again $l(a_m)$ should be less than $l(a_j) + l(a_l) + l(a_h)$.

Obviously this process can continue if there is a large variation in the codeword lengths. Therefore, this approach becomes cumbersome for moderately large codebooks.

A somewhat different way of looking at this issue, suggested in a slightly different context by Massey [37], is to block the channel output bit stream into fixed length words where the fixed length is longer than the longest binary codeword in the channel input. Then, the path metric becomes the logarithm of

$$\frac{P(\hat{y}_i = \tilde{r} \mid y_i)P(y_i \mid y_{i-1})}{\Sigma_l P(\hat{y}_i = \tilde{r} \mid y_i = a_l)P(y_i = a_l \mid y_{i-1})} \tag{11}$$

where $\tilde{r}$ denotes the word corresponding to a received block of bits. While there are some complications here as well, in the interpretation of $P(\hat{y}_i \mid y_i)$, the main difficulty is a computational one. The simplest implementation of the JSC decoder requires that the path metrics be stored in a lookup table. In the case of identical input and output alphabets of size M, the lookup table size is M^3. However, with this approach, the lookup table size is $M^2 2^{L+1}$ where L is the longest codeword. This exponential increase with even moderate codeword lengths makes this approach impractical at least for a lookup table implementation. An implementation which does not use a lookup table, and instead computes the path metric at each step may still be possible with special purpose dedicated hardware.

Given the difficulties involved with implementation of the exact path metric of the MAP JSC decoder, we have proposed two approximations which provide a high level of error protection while being computationally simple and easy to implement. First consider (4). We approximate the denominator as

$$P(\hat{y}_i = \hat{r} \mid y_{i-1}) \cong P(\hat{y}_i = \hat{r}).$$

and therefore the entire expression as

$$P(y_i = a_j \mid \hat{y}_i = \hat{r}, y_{i-1} = a_m) \qquad (12)$$

$$\cong \frac{P(\hat{y}_i = \hat{r} \mid y_i = a_j) P(y_i = a_j \mid y_{i-1} = a_m)}{P(\hat{y}_i = \hat{r})}$$

where the number of bits in $\hat{r}$ is the number of bits used to represent a_j. The denominator is further approximated by assuming equally likely reception of bits as

$$P(\hat{y}_i = \hat{r}) = (\tfrac{1}{2})^{l(a_j)} \qquad (13)$$

where $l(a_j)$ is the number of bits in a_j and therefore in $\hat{r}$. The computation of the path metric then proceeds as follows: the conditional probability $P(y_i = a_j \mid y_{i-1} = a_{j-1})$ is read from a lookup table and the transition probability is computed by assuming a binary symmetric channel with known crossover probability. This form of the path metric is easy to implement and the simulation results of Section V show the scheme to be highly effective.

An even simpler approximation is to use the Hamming distance between the received bits and the candidate sequence elements as the branch and path metric. Of course the candidate sequence elements are selected from allowed sequence values. (Recall that the π operator, by construction, disallows certain sequences.) We present results using this metric in Section V. This approximation causes a drop in performance from about a half dB in the low noise region to about 1.5 to 2 dB in the high noise region. Given the simplicity of implementation for this scheme, this may very well be an acceptable cost.

Once the path metric has been obtained, the decoder structure needs to be elucidated. We do so in the next section.

IV. Decoder Structure

The form of the path metric in (1) is a familiar one and several decoder structures exist which maximize (or minimize) additive path metrics of this form. One of the most popular ones is the Viterbi decoder structure. Recall that the Viterbi decoder limits the total number of candidate paths (solutions) to some finite number M where M is the number of different values a solution can take at any given time increment. This is done by using a trellis structure that only includes allowed paths or transitions. For the problem considered here M would be the size of the output alphabet of the π operator. In most applications where the Viterbi decoder is used, the codewords are of fixed length and therefore the candidate paths are of the same length. This is not true in the current case. However, this problem can be resolved rather simply by associating a pointer with each candidate path. The pointer counts the number of bits used to form the path it is associated with.

To see how this works consider the following example. Let the input alphabet to the π operator be of size two; $S = \{s_0, s_1\}$. Suppose the input sequence to the π operator is

$$s_0\ s_0\ s_0\ s_1\ s_0\ s_0$$

then the output of the π operator will be

$$a_0\ a_0\ a_1\ a_2\ a_0$$

If the Huffman code for the π operator output is

$$a_0{:}0,\ a_1{:}10,\ a_2{:}110,\ a_3{:}111$$

then the transmitted binary sequence will be

0 0 1 0 1 1 0 0

Suppose there is an error in the fourth bit and the received sequence is

0 0 1 1 1 1 0 0

The decoder operation is shown in Figure 1, where the metric being used is the Hamming distance. The branches are labelled with a pair of numbers. The first number is the accumulated number of bits used by the path that includes that branch while the second number is the Hamming distance between the received bits and the candidate solution. The receiver assumes a starting value of a_0. In the first step there are two possibilities, that the transmitted word was a_0 or a_1. If we assume the transmitted word was a_0 we use up one bit and the Hamming distance is zero. If a_1 is assumed then we use two bits and the Hamming distance is one. Therefore, the lower branch (to a_0) is labelled 1,0 while the branch to a_1 is labelled 2,1. This procedure is continued with conflicts being resolved by picking the path with the lower Hamming distance. The procedure is shown in Figure 1.

V. Simulation Results

The techniques presented in this paper were applied to an image coding scheme. The data compression scheme was a DPCM system with a fixed four level nonuniform Max quantizer and a one-tap predictor. The data compaction scheme is a sixteen-level Huffman coder. The average rate for this system was 2.3 bits per pixel. End of line resynchronization is assumed for the receiver. A block diagram of the system is shown in Figure 2.

The performance with both metrics is shown in Figure 3 and Figure 4. Both figures plot the same results where Figure 3 emphasizes the performance in the low noise region and Figure 4 emphasizes performance at high channel error rates. The curves are labeled "Approx 1," "Approx 2," and "No Protection." The curve labeled Approx 1 is the performance curve for the system which uses the metric approximation of (12) and (13). The curve labeled Approx 2 is the system which uses the second approximation, i.e., the Hamming distance between the received bits and the candidate sequence elements. The curve labeled "No Protection" is the system without the joint source/channel coding scheme. Both metric approximations provide a high degree of protection for low to moderate channel error rates. At high channel error rates, while both the systems provide substantial performance improvements over the unprotected system, the system with the Hamming distance metric provides lower performance than the system with the approximation of (12) and (13). However, as mentioned before, this might be a small cost to pay for the simplicity of implementation.

REFERENCES

[1] C.E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379-432, July 1948.

[2] T. Berger, *Rate Distortion Theory*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1971.

[3] J.L. Massey, "Joint Source and Channel Coding," in *Communication Systems and Random Process Theory*, J.K. Skwirzynski, ed., Sijthoff and Nordhoff: Netherlands, pp. 279-293, 1978.

[4] T.C. Ancheta, Jr., "Bounds and Techniques for Linear Source Coding," Ph.D. dissertation, Dept. of Elec. Eng., Univ. of Notre Dame, Aug. 1977.

[5] J.G. Dunham and R.M. Gray, "Joint source and channel trellis encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 516-519, July 1981.

[6] E. Ayanoglu and R.M. Gray, "The design of joint source and channel trellis waveform coders," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 855-865, Nov. 1987.

[7] J.W. Modestino and D.G. Daut, "Combined source-channel coding of images," *IEEE Trans. Commun.*, vol. COM-27, pp. 1644-1659, Nov. 1979.

[8] J.W. Modestino, D.G. Daut and A.L. Vickers, "Combined source-channel coding of images using the block cosine transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1262-1274, Sept. 1981.

[9] J.W. Modestino, V. Bhaskaran, and J.B. Anderson, "Tree encoding of images in the presence of channel errors," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 677-697.

[10] D. Comstock and J.D. Gibson, "Hamming coding of DCT compressed images over noisy channels," *IEEE Trans. Commun.*, vol. COM-32, pp. 856-861, July 1984.

[11] C.C. Moore and J.D. Gibson, "Self-orthogonal convolutional coding for the DPCM-AQB speech encoder," *IEEE Trans. Commun.*, vol. COM-32, pp. 980-982, Aug. 1984.

[12] R.C. Reininger, "Backward adaptive lattice and transversal predictors in ADPCM," *IEEE Trans. Commun.*, vol. COM-33, pp. 74-82, Jan. 1985.

[13] D.J. Goodman and C.E. Sundberg, "Combined source and channel coding for variable bit rate speech transmission," *Bell Syst. Tech. J.*, vol. 62, pp. 2017-2036, Sept. 1983.

[14] D.J. Goodman and C.E. Sundberg, "Transmission errors and forward error correction in embedded differential pulse code modulation," *Bell Syst. Tech. J.*, vol. 62, pp. 2735-2764, Nov. 1983.

[15] J.D. Gibson and R.C. Reininger, "Forward and Backward Prediction in ADPCM Over Nonideal Channels," *Conf. Rec.*, Int. Conf. Commun., June 8-12, 1980, Seattle, WA, pp. 42.5.1-42.5.5.

[16] A.J. Kurtenbach and P.A. Wintz, "Quantizing for noisy channels," *IEEE Trans. Commun. Technol.*, vol. COM-17, pp. 291-302, April 1969.

[17] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: An approach to combined source-channel coding," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 827-838, Nov. 1987.

[18] V. Vaishampayan and N. Farvardin, "Optimal block cosine transform image coding for noisy channels," submitted to *IEEE Trans. Commun.*

[19] H. Kumazawa, M. Kashahara, and T. Namekawa, "A construction of vector quantizers for noisy channels," *Electronics and Engineering in Japan*, vol. 67-B, no. 4, pp. 39-47, 1984.

[20] N. Farvardin, "A Study of Vector Quantization for Noisy Channels," Tech. Res. Rpt., SRC TR 88-15, Univ. of Maryland, College Park, MD.

[21] V. Vaishampayan and N. Farvardin, "The Joint Design of Block Source Coders and Modulation Signal Sets," Twenty-Second Annual Conf. of Information Sciences and Systems, March 16-18, 1988, Princeton Univ., Princeton, NJ.

[22] K-Y Chang and R.W. Donaldson, "Analysis, optimization, and sensitivity study of differential PCM systems operating on noisy communication channels," *IEEE Trans. Commun.*, vol. COM-20, pp. 338-350, June 1972.

[23] N. Rydbeck and C.W. Sundberg, "Analysis of digital errors in nonlinear PCM systems," *IEEE Trans. Commun.*, vol. COM-24, pp. 59-65, Jan. 1976.

[24] J.R.B. DeMarca and N.S. Jayant, "An Algorithm for Assigning Binary Indices to the Codevectors of a Multi-dimensional Quantizer," *Proc.* IEEE Int. Commun. Conf., Seattle, WA, pp. 1128-1132, June 1987.

[25] J.-H. Chen, G. Davidson, A. Gersho and K. Zeger, "Speech Coding for the Mobile Satellite Experiment," *Proc.*, IEEE Int. Comm. Conf., Seattle, WA, pp. 756-763, June 1987.

[26] K.A. Zeger and A. Gersho, "Zero redundancy channel coding in vector quantization," IEEE Electronics Letters, vol. 23, pp. 654-655, June 1987.

[27] R. Steele, D.J. Goodman, and C.A. McGonegal, "Partial correction of transmission errors in DPCM without recourse to error correction coding," *Elec. Lett.*, vol. 13, pp. 351-353, June 1977.

[28] R. Steele, D.J. Goodman, and C.A. McGonegal, "A difference detection and correction scheme for combatting DPCM transmission errors," *IEEE Trans. Commun.*, vol. COM-27, pp. 252-255, Jan. 1979.

[29] K.N. Ngan and R. Steele, "Enhancement of PCM and DPCM images corrupted by transmission errors," *IEEE Trans. Commun.*, vol. COM-30, pp. 257-269, Jan. 1982.

[30] R.C. Reininger and J.D. Gibson, "Soft decision demodulation and transform coding of images," *IEEE Trans. Commun.*, vol. COM-31, pp. 572-577, April 1983.

[31] K. Sayood and J.C. Borkenhagen, "Utilization of Correlation in Low Rate DPCM Systems for Channel Error Protection," *Proceedings*, IEEE International Conference on Communications, June 1986, pp. 1888-1892.

[32] K. Sayood and J.C. Borkenhagen, "Use of residual redundancy in the design of joint source/channel coders," submitted to *IEEE Transactions on Communications*.

[33] M.E. Hellman, "On using natural redundancy for error detection," *IEEE Trans. Commun.*, vol. COM-22, pp. 1690-1693, Oct. 1974.

[34] M.E. Hellman, "Convolutional source encoding," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 651-656, Nov. 1975.

[35] K. Sayood and J.D. Gibson, "Maximum a posteriori joint source/channel coding," Twenty-Second Annual Conf. on Information Sciences and Systems, March 16-18, Princeton Univ., Princeton, NJ.

[36] R.E. Blahut, *Principles and Practice of Information Theory*, Addison-Wesley, 1987.

[37] J.L. Massey, "Variable Length Codes and the Fano Metric," *IEEE Trans. Inform. Theory*, Jan. 1972.

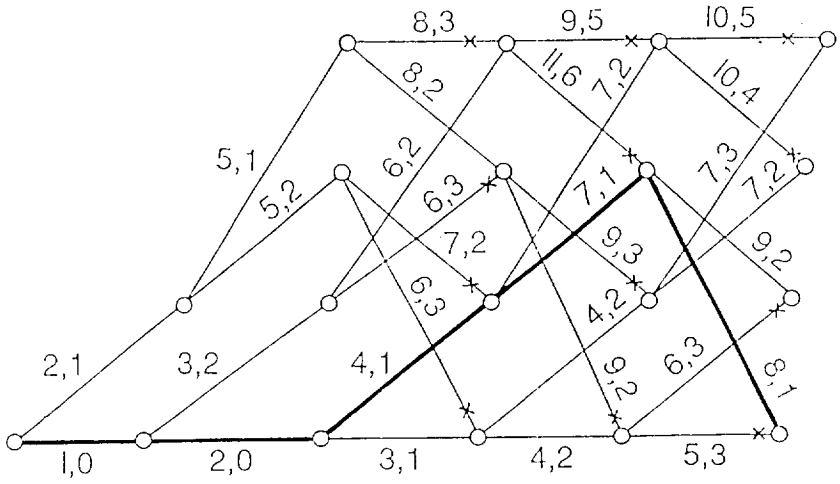


Figure 1. Decoding procedure for variable length codes

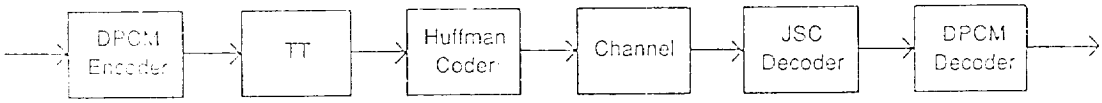


Figure 2. Proposed system

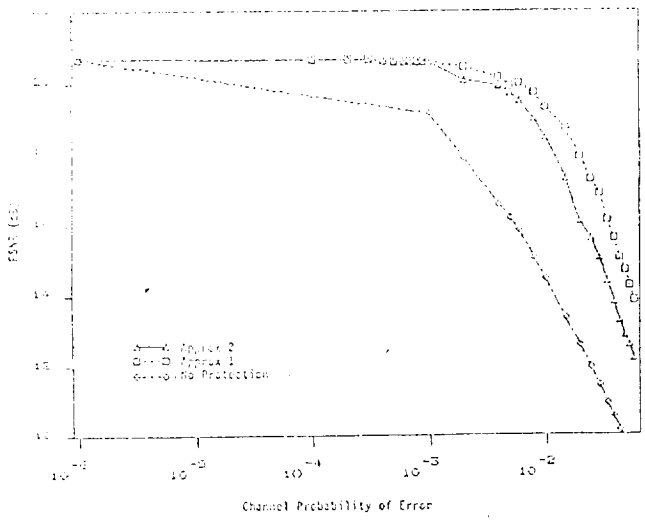


Figure 3. Comparison of performance with different decoder metrics

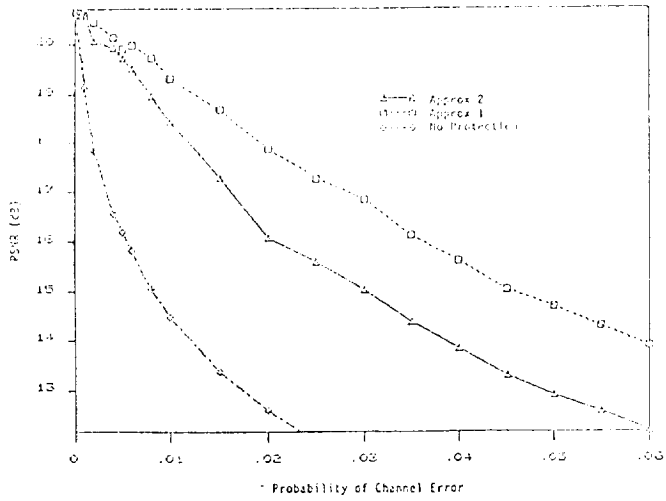


Figure 4. Comparison of performance with different decoder metrics

Appendix 2- Item 4

Presented at the 25th Asilomar Conference on Signals, Systems, and Computers.

A Joint Source/Channel Coder Design*

Fuling Liu and Khalid Sayood
Dept. of Electrical Engineering
and the Center for Communication
& Information Science
Univ. of Nebraska, Lincoln, NE 68588-0511

Jerry D. Gibson†
Information Systems Laboratory
and the Telecommunications Program
Dept. of Electrical Engineering
Stanford Univ., Stanford, CA 94305

Abstract

Source coders and channel coders are generally designed separately without reference to each other. This approach is justified by a famous result of Shannons. However, there are many situations in practice in which the assumptions upon which this result is based are violated. Specifically, we examine the situation where there is residual redundancy at the source coder output. We have previously shown that this residual redundancy can be used to provide error correction using a Viterbi decoder. In this paper we present the second half of the design; the design of encoders for this situation. We show through simulation results that the proposed coders consistently outperform conventional source-channel coder pairs with gains of up to 12dB at high probability of error.

1 Introduction

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be treated separately without any loss of performance for the overall system [1]. The basic design procedure is to select a source encoder which changes the source sequence into *iid* bits followed by a channel encoder which encodes the bits for reliable transmission over the channel. However, the separation argument no longer holds if either of the following two situations occur:

i. The input to the source decoder is different from the output of the source encoder, which happens when the link between the source encoder and source decoder is no longer error free, or

ii. The source coder output contains redundancy.

Case (i) occurs when the channel coder does not achieve zero error probability and case (ii) occurs when the source encoder is suboptimal. These two situations are common occurrences in practical systems where source or channel models are imperfectly known, complexity is a serious issue, or significant delay is not tolerable. Approaches developed for such situations are usually grouped under the general heading of joint source/channel coding.

Most joint source channel coding approaches can be classified in two main categories; (A) approaches which entail the modification of the source coder/decoder structure to reduce the effect of channel errors [2-10], and (B) approaches which examine the distribution of bits between the source and channel coders [11, 12]. The first set of approaches can be divided still further into two classes. One class of approaches examines the modification of the overall structure [2-5], while the other deals with the modification of the decoding procedure to take advantage of the redundancy in the source coder output [6-10].

In this paper we present an approach to joint source/channel coder design, which belongs to category A, and hence we explore a technique for designing joint source/channel coders, rather than ways of distributing bits between source coders and channel coders. We assume that the two nonideal situations referred to earlier are present. For a nonideal source coder, we use MAP arguments to design a decoder which takes advantage of redundancy in the source coder output to perform error correction. We then use the decoder structure to infer the encoder design.

2 The Design Criterion

For a discrete memoryless channel (DMC), let the channel input alphabet be denoted by A =

*This work was supported in part by NASA Lewis Research Center (NAG 3-806) and NASA Goddard Space Flight Center (NAG 5-916)

†On leave from Dept. of Electrical Engr. Texas A&M Univ.

$\{a_0, a_1, \ldots, a_{M-1},\}$, and the channel input and output sequences by $Y = \{y_0, y_1, \ldots, y_{L-1}\}$ and $\hat{Y} = \{\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{L-1}\}$, respectively. If $\mathcal{A} = \{A_i\}$ is the set of sequences $A_i = \{\alpha_{i,0}, \alpha_{i,1}, \ldots, \alpha_{i,L-1}\}$, $\alpha_{i,k} \epsilon A$, then the optimum receiver (in the sense of maximizing the probability of making a correct decision) maximizes $P[C]$, where

$$P[C] = \sum_{A_i} P[C|\hat{Y}]P[\hat{Y}].$$

This in turn implies that the optimum receiver maximizes $P[C|\hat{Y}]$. When the receiver selects the output to be A_k, then $P[C|\hat{Y}] = P[Y = A_k|\hat{Y}]$. Thus, the optimum receiver selects the sequence A_k such that

$$P[Y = A_k|\hat{Y}] \geq P[Y = A_i|\hat{Y}] \ \forall_i.$$

Lemma 1

Let y_i be the input to a DMC. Given y_{i-1}, y_i is conditionally independent of $y_{n-k}, k > 1$. If $\hat{y}_0 = y_0$ then the optimum receiver selects a sequence A_i to maximize $\Pi_{i=1}^{L-1} P(y_i|y_{i-1}, \hat{Y}_i)$ where $\hat{Y}_k = \{\hat{y}_k, \hat{y}_{k+1}, \ldots, \hat{y}_{L-1}\}$.

The lemma addresses the situation in case (ii), i.e., the situation in which the source coder output (which is also the channel input sequence) contains redundancy. Using this lemma, we can design a decoder which will take advantage of dependence in the channel input sequence. The lemma provides the mathematical structure for the decoder. The physical structure can be easily obtained by examining the quantity to be maximized. The optimum decoder maximizes $P(Y|\hat{Y})$ or equivalently $\log P(Y|\hat{Y})$, but

$$\log P(Y|\hat{Y}) = \sum \log P(y_i|\hat{Y}_i, y_{i-1}) \qquad (1)$$

which is similar in form to the path metric of a convolutional decoder. Error correction using convolutional codes is made possible by explicitly limiting the possible codeword to codeword transitions, based on the previous code input and the coder structure. In this case, while there is no structure being imposed by the encoder, there is sufficient residual structure in the source coder output that can be used for error correction. This structure can be quantified in light of the Lemma. That is, the structure is reflected in the conditional probabilities, and can be utilized via the path metric in (1) in a decoder similar in structure to a convolutional decoder. However, to implement this decoder we need to be able to compute the path metric. Unfortunately the quantity $P(y_i|\hat{Y}_i, y_{i-1})$ is difficult to estimate. We have therefore used various approximations to this quantity with some success. In [8, 9] $P(y_i|\hat{Y}_i, y_{i-1})$ is approximated by $P(y_i|\hat{y}_i, y_{i-1})$ with excellent results. Other approximations can be found in [13].

In [9] we showed that the use of the decoder led to dramatic improvements under high error rate conditions. However at low error rates the performance improvement was from nonexistent to minimal. This is in contrast to standard error correcting approaches, in which the greatest performance improvements are at low error rates, with a rapid deterioration in performance at high error rates. In this work we combine the two approaches to develop a joint source channel codec which provides protection equal to the standard channel encoders at low error rates while providing significant error protection at high error rates.

3 Proposed Encoder Structure

In the conventional error protection approach we introduce structure in the transmitted bitstream. In the approach proposed in [9], we use the residual structure in the (generally nonbinary) source coder output sequence. To combine the two approaches, we need to introduce additional structure without disturbing the structure already present. Because of the nature of the decoding approach, a convolutional encoder would be most appropriate for introducing structure. However, a standard binary convolutional encoder will tend to destroy the structure in the source coder output. To preserve the residual structure while introducing additional structure we propose to use nonbinary convolutional encoders (NCE) whose input alphabet is the output alphabet of the source coder.

Let x_n, the input to the NCE, be selected from the alphabet $A = \{0, 1, 2, ..., N-1\}$, and let y_n, the output alphabet of the NCE, be selected from the alphabet $S = \{0, 1, 2, ..., M-1\}$. Then, two of the proposed NCEs can be described by the following mappings

1. $M = N^2$; $y_n = Nx_{n-1} + x_n$

The number of bits required to represent the output alphabet using a fixed length code is

$$\lceil \log_2(M) \rceil = \lceil \log_2(N^2) \rceil = \lceil 2 \log_2(N) \rceil$$

Therefore in terms of rate, this coder is equivalent to a rate 1/2 convolutional encoder. The encoder memory in bits is $2\lceil \log_2(N) \rceil$ as each output value depends on two input values.

As an example, consider the situation when $N = 4$. Then $A = \{0, 1, 2, 3\}$ and $S = \{0, 1, 2, ..., 15\}$. Given

the input sequence x_n : 0 1 3 0 2 1 1 0 3 3 and assuming the encoder is initialized with zeros, the output sequence will be y_n : 0 1 7 12 2 9 5 4 3 15.

The encoder memory is four bits. Notice that while the encoder output alphabet is of size N^2, at any given instant the encoder can only emit one of N different symbols as should be the case for a rate 1/2 convolutional encoder. For example if $y_{n-1} = 0$, then y_n will take on a value from $\{0, 1, 2, ..., (N-1)\}$. In general, given a value for y_{n-1}, y_n will take on a value from $\{\alpha N, \alpha N + 1, \alpha N + 2, ..., \alpha N + N - 1\}$, where $\alpha = y_{n-1}(mod N)$. This structure can be used by the decoder to provide error protection. The encoder is shown in Figure 1a.

2. $M = N^3$; $y_n = N^2 x_{2n-2} + N x_{2n-1} + x_{2n}$

The final encoder we consider is equivalent to a rate 2/3 convolutional coder. Notice that while the input output relationship looks similar to a rate 1/3 encoder, we generate one output for every two inputs. Thus, while the number of bits needed to represent one letter from the output alphabet is three times the bits needed to represent a letter from the input alphabet, because two input letters are represented by a single output letter, the rate is 2/3. Again, assuming a value of 4 for N, the output alphabet is of size 64, and for the input sequence used previously, the output sequence is y_n : 0 52 35 22 49 3.

The encoder memory is again 6 bits. A block diagram of the encoder is shown in Figure 1b. The rate of the encoder can also be inferred from the fact that while the encoder output alphabet is of size N^3, at any instant the encoder can transmit one of N^2 (instead of N) symbols. Given a value for y_{n-1}, y_n can take on a value from the alphabet $\{\gamma N^2, \gamma N^2 + 1, ..., \gamma N^2 + (N^2 - 1)\}$ where $\gamma = y_{n-1}(mod N)$.

4 Binary Encoding of the NCE Output

We will make use of the residual structure in the source coder output (which is preserved in the NCE output) at the receiver. However, we can also make use of this structure in selecting binary codes for the NCE output. An intelligent assignment of binary codes can improve the error correcting performance of the system. Our strategy is to try to maximize the Hamming distance between codewords that are likely to be mistaken for one another.

First we obtain a partition of the alphabet based on the fact that given a particular value for y_{n-1}, y_n can only take on values from a subset of the full alphabet. To see this, consider the rate 1/2 NCE; then the alphabet S can be partitioned into the following sub-alphabets:

$$S_j = (jN, jN + 1, ..., jN + N - 1) \quad j = 0, 1, ...N - 1$$

where the encoder will select letters from alphabet S_j at time n if $j = y_{n-1}(\text{mod}N)$. Now for each sub-alphabet we have to pick N codewords out of M $(= N^2)$ possible choices. We first pick the sub-alphabet containing the most likely letter. The letters in the sub-alphabet are ordered according to their probability of occurrence. We assign a codeword a from the list of available codewords to the most probable symbol. Then, assign the complement of a to the next symbol on the list. Therefore the distance between the two most likely symbols in the list is $K = \lceil \log_2 M \rceil$ bits. We then pick a codeword b from the list which is at a Hamming distance of $K/2$ from a and assign it and its complement to the next two elements on the list. This process is continued with the selection of letters that are$K/2^k$ away from a at the k^{th} step until all letters in the subalphabet have a codeword assigned to them. We then pick the sub-alphabet that contains the next most likely letter. It is assigned the available codeword at maximum distance from a. The procedure for assigning codewords within the sub-alphabet is then repeated. The assignment for a rate 1/2 with $N = 4$ code is shown in Table 1.

5 Simulation Results

The proposed approach was simulated using a two-bit DPCM system as the source coder, and the three NCE described in section 3. The source used were standard test images USC Girl, USC Couple and a 256x256 portion of Lena. The decoder structure used was that of a Viterbi decoder with branch metric $\log L$

$$L = \frac{P(\hat{y}_i \mid y_i)\, P(y_i \mid y_{i-1}, y_{i-2})}{P(\hat{y}_i)}$$

where y_i denotes the NCE output and $\hat{y}_i$ denotes the corrupted channel output. The probabilities $P(y_i \mid y_{i-1}, y_{i-2})$ were estimated using a training sequence. This requires estimating MN^2 probabilities, which were estimated using the USC Girl image. The test images were the USC Couple and Lena images.

The proposed scheme was compared with a conventional source coder-convolutional coder combination. The source coder and source sequence were the same in both systems. The convolutional codes selected were the codes with maximal d_{free} and the

same rate and memory characteristics as the proposed NCEs from [14]. The performance measure was the signal-to-noise-ratio (SNR) defined as

$$SNR = 10 \log_{10} \frac{\sum u_i^2}{\sum (u_i - \hat{u}_i)^2}$$

where u_i is the input to the source encoder and $\hat{u}_i$ is the output of the source decoder.

The results show consistent improvement in performance for the proposed system. At low probabilities of error both systems perform very well. At high probabilities of error ($> 10^{-2}$), however, there is a substantial improvement in performance when the proposed system is used.

In Figures 2a and 2b we show the results of one of the simulations for the rate 1/2 codes. The binary assignment of Table 1 was used in the simulation. Notice the flatness of the performance curve for the proposed system. While the proposed system consistently outperforms the conventional system, it is at higher probabilities of error that the differences really become significant. At a probability of error of 10^{-1} there is almost a 6dB difference in the performance of the two systems! This "flattening out" of the performance curve makes the approach useful for a large variety of channel error conditions.

Similar performance improvements can be seen for the rate 2/3 system of the second mapping. The performance curves are shown in Figure 3. Notice that again the proposed system consistently outperforms the conventional system. In this case at a probability of error of 10^{-1} the performance improvement is more than 12dB! In fact, the proposed rate 2/3 system performs better than the conventional rate 1/2 system.

6 Conclusion

If the source and channel coder are designed in a "joint" manner, that is the design of each takes into account the overall conditions (source as well as channel statistics), we can obtain excellent performance over a wide range of channel conditions. In this paper we have presented one such design. The resulting performance improvement seems to validate this approach.

References

[1] C. E. Shannon. *Bell Syst. Tech. J.* 27:379–423, 623–656. 1948.

[2] E. Ayanoğlu and R. M. Gray. *IEEE Trans. Inform. Theory.* IT-33:855–865. Nov. 1987.

[3] T. C. Ancheta, Jr. Ph.D. dissertation, Dept. of Electrical Engr., Univ. of Notre Dame. Aug. 1977.

[4] K–Y Chang and R. W. Donaldson. *IEEE Trans. Commun.* COM-20:338–350. June 1972.

[5] D. J. Goodman and C. E. Sundberg. *Bell Syst. Tech. J.* 62:2017–203. Sept. 1983.

[6] R. C. Reininger and J. D. Gibson. *IEEE Trans. Commun.* COM-31:572–577. April 1983.

[7] R. Steele et al. *IEEE Trans. Commun.* COM-27:252–255. Jan. 1979.

[8] K. Sayood and J. C. Borkenhagen. *Proceedings IEEE ICC '86.* 1888–1892. June 1986.

[9] K. Sayood and J. C. Borkenhagen. *IEEE Transactions on Communications.* COM-39. June 1991.

[10] K. Sayood and J. D. Gibson. *Proc. 22nd Annual CISS*, Princeton, NJ. 380-385. Mar. 1988.

[11] J. W. Modestino et al. *IEEE Trans. Commun.* COM-29:1262–1274. Sept. 1981.

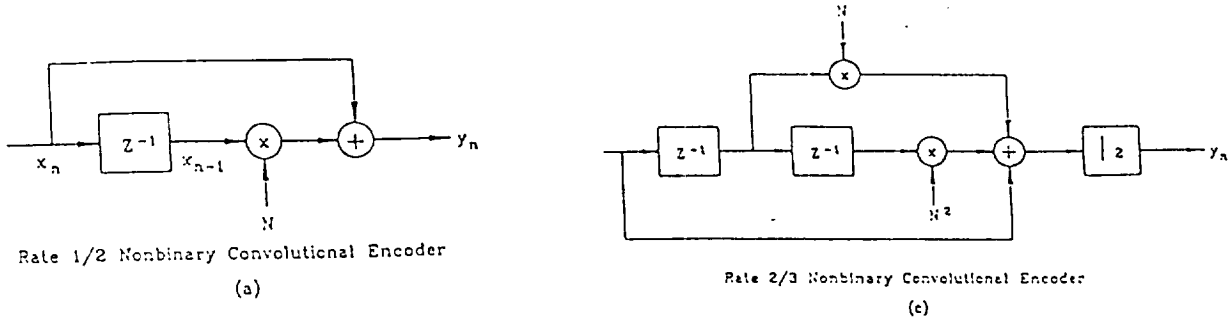
[12] D. Comstock and J. D. Gibson. *IEEE Trans. Commun.* COM-32:856–861. July 1984.

[13] K. Sayood, J. D. Gibson, and F. Liu. *Proc. 22nd Annual Asilomar Conference on Circuits, Systems, and Computers.* 102-106. Nov. 1988.

[14] S. Lin and D. J. Costello. *Error Control Coding.* Prentice Hall. 1983.

Table 1: Codeword Assignments

Symbol	Code	Symbol	Code
0	0000	8	1011
1	0011	9	0111
2	1100	10	0100
3	1111	11	1000
4	1110	12	0101
5	1101	13	1001
6	0001	14	1010
7	0010	15	0110



Rate 1/2 Nonbinary Convolutional Encoder

(a)

Rate 2/3 Nonbinary Convolutional Encoder

(c)

Figure 1. Proposed Nonbinary Convolutional Encoders.

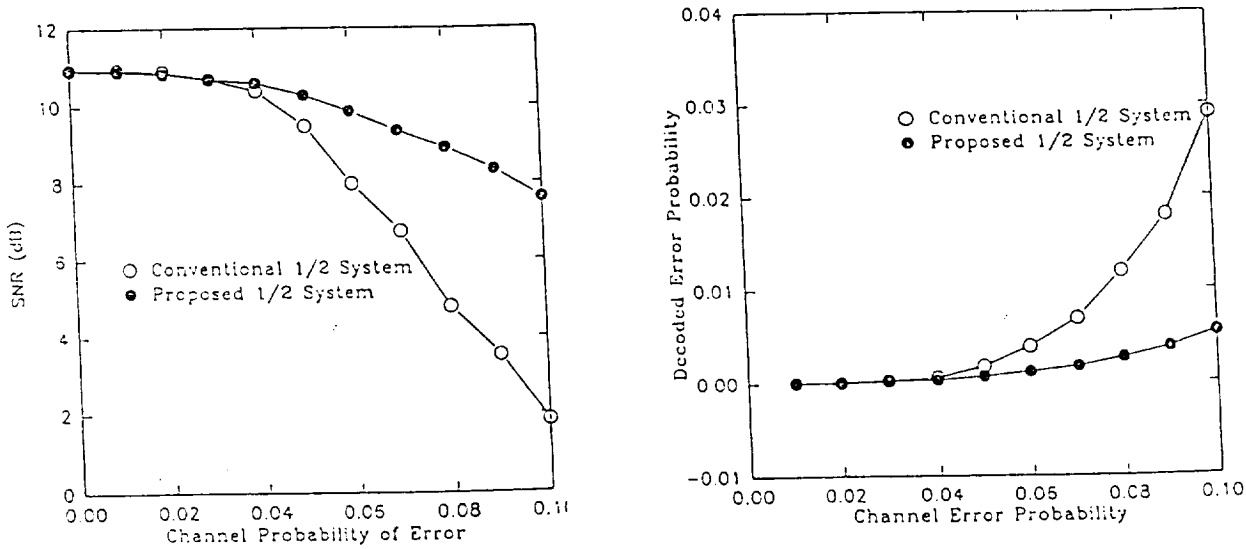


Figure 2. Performance Comparison of rate 1/2 Systems.

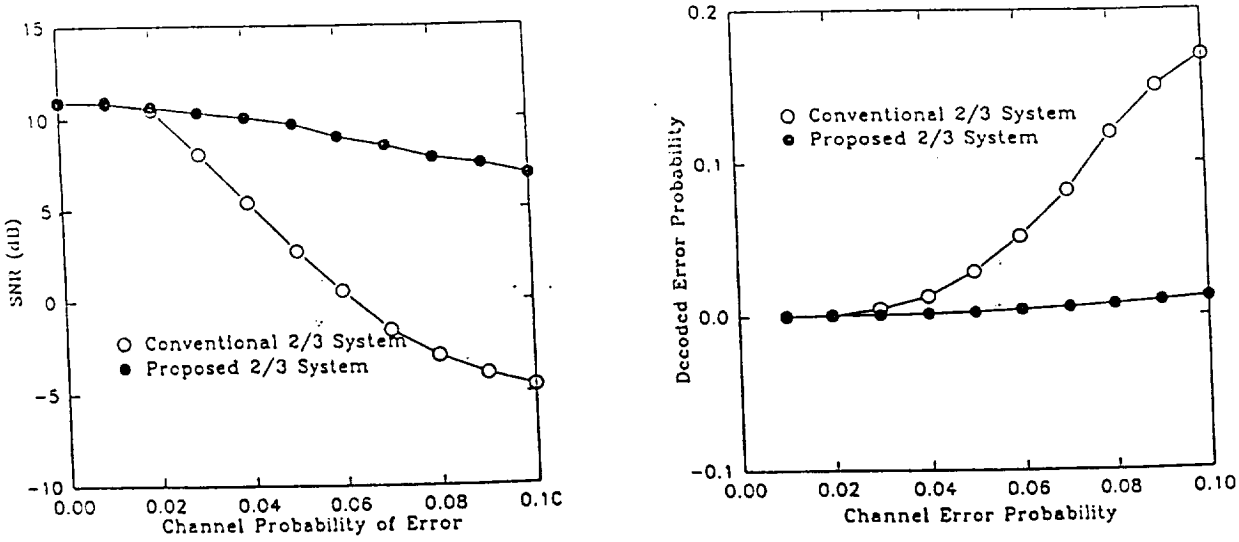


Figure 3. Performance Comparison of rate 2/3 Systems.

Appendix 2- Item 5

Draft

A Joint Source/Channel Coder Design*

Fuling Liu and Khalid Sayood
Dept. of Electrical Engineering
and the Center for Communication
& Information Science
Univ. of Nebraska, Lincoln, NE 68588-0511

Jerry D. Gibson
Dept. of Electrical Engineering
Texas A&M University
College Station, TX 97843

Abstract

Source coders and channel coders are generally designed separately without reference to each other. This approach is justified by a famous result of Shannons. However, there are many situations in practice in which the assumptions upon which this result is based are violated. Specifically, we examine the situation where there is residual redundancy at the source coder output. We have previously shown that this residual redundancy can be used to provide error correction using a Viterbi decoder. In this paper we present the second half of the design; the design of encoders for this situation. We show through simulation results that the proposed coders consistently outperform conventional source-channel coder pairs with gains of up to 12dB at high probability of error.

1 Introduction

One of Shannon's many fundamental contributions was his result that source coding and channel coding can be treated separately without any loss of performance for the overall system [1]. The

*This work was supported in part by NASA Lewis Research Center (NAG 3-806) and NASA Goddard Space Flight Center (NAG 5-916)

basic design procedure is to select a source encoder which changes the source sequence into a series of independent, equally likely binary digits followed by a channel encoder which accepts binary digits and puts them into a form suitable for reliable transmission over the channel. However, the separation argument no longer holds if either of the following two situations occur:

i. The input to the source decoder is different from the output of the source encoder, which happens when the link between the source encoder and source decoder is no longer error free, or

ii. The source coder output contains redundancy.

Case (i) occurs when the channel coder does not achieve zero error probability and case (ii) occurs when the source encoder is suboptimal. These two situations are common occurrences in practical systems where source or channel models are imperfectly known, complexity is a serious issue, or significant delay is not tolerable. Approaches developed for such situations are usually grouped under the general heading of joint source/channel coding.

Most joint source channel coding approaches can be classified in two main categories; (A) approaches which entail the modification of the source coder/decoder structure to reduce the effect of channel errors, and (B) approaches which examine the distribution of bits between the source and channel coders. The first set of approaches can be divided still further into two classes. One class of approaches examines the modification of the overall structure, while the other deals with the modification of the decoding procedure to take advantage of the redundancy in the source coder output.

To the first class belongs the work of Dunham & Gray [2] who proved the existence of joint source channel trellis coding systems for certain fidelity criteria, and a design of a joint source channel trellis coder presented by Ayanoğlu and Gray [3], where the design procedure is the generalized Lloyd algorithm. Further, Massey [4] and Ancheta [5] showed that for distortionless transmission

of the source using linear joint source channel encoders, equivalent performance can be obtained with a significant reduction in complexity. Chang and Donaldson [6] propose modifications to the DPCM system to reduce the effect of channel errors, while Kurtenbach and Wintz [7] and Farvardin and Vaishampayan [8] study the problem of optimum quantizer design for noisy channels. Goodman and Sundberg [9,10] propose an embedded DPCM system which consists of a two bit DPCM and a two bit PCM system in parallel.

In the second class of category A, we include the work of Reininger and Gibson [11], who use the fact that coefficients in neighboring blocks in a transform coding scheme will not vary greatly, and thus use coefficients from neighboring blocks to correct a possible error, and the work of Steele, Goodman and McGonegal [12,13], who propose a difference detection and correction scheme for broadcast quality speech. In this scheme the receiver infers an error whenever an individual sample to sample difference is greater than the mean squared difference of a 21 sample sliding block. When an error is detected, the received sample is replaced by the output of a smoothing circuit. Ngan and Steele [14] use a similar method for recovering from errors in an image transmission system. Sayood and Borkenhagen [16, 17] use the redundancy at the source coder output to perform sequence estimation. Sayood and Gibson [18] examine "desirable" properties for encoders which enhance sequential estimation performance.

The work of Modestino, Daut and Vickers [19] belongs to category B. In their study of transform coding they examine tradeoffs between allocating bits for source and channel coding. Comstock and Gibson [20] extend this work and provide an explicit mechanism for allocating bits between a source coder and a Hamming channel coder. Additionally, Moore and Gibson [21] study the allocation of bits between a DPCM coder and self orthogonal convolutional coding.

In this paper we present an approach to joint source/channel coder design, which belongs to category A, and hence we explore a technique for designing joint source/channel coders, rather

than ways of distributing bits between source coders and channel coders. We assume that the two nonideal situations referred to earlier are present. For a nonideal source coder, we use MAP arguments to design a decoder which takes advantage of redundancy in the source coder output to perform error correction. We then use the decoder structure to infer the encoder design.

2 The Design Criterion

For a discrete memoryless channel (DMC), let the channel input alphabet be denoted by $A = \{a_0, a_1, \ldots, a_{M-1},\}$, and the channel input and output sequences by $Y = \{y_0, y_1, \ldots, y_{L-1}\}$ and $\hat{Y} = \{\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{L-1}\}$, respectively. If $\mathcal{A} = \{A_i\}$ is the set of sequences $A_i = \{\alpha_{i,0}, \alpha_{i,1}, \ldots, \alpha_{i,L-1}\}$, $\alpha_{i,k} \epsilon A$, then the optimum receiver (in the sense of maximizing the probability of making a correct decision) maximizes $P[C]$, where

$$P[C] = \sum_{A_i} P[C|\hat{Y}]P[\hat{Y}].$$

This in turn implies that the optimum receiver maximizes $P[C|\hat{Y}]$. When the receiver selects the output to be A_k, then $P[C|\hat{Y}] = P[Y = A_k|\hat{Y}]$. Thus, the optimum receiver selects the sequence A_k such that

$$P[Y = A_k|\hat{Y}] \geq P[Y = A_i|\hat{Y}] \ \forall_i.$$

Lemma 1

Let y_i be the input to a DMC. Given y_{i-1}, y_i is conditionally independent of $y_{n-k}, k > 1$. If $\hat{y}_0 = y_0$ then the optimum receiver selects a sequence A_i to maximize $\Pi_{i=1}^{L-1} P(y_i|y_{i-1}, \hat{Y}_i)$ where $\hat{Y}_k = \{\hat{y}_k, \hat{y}_{k+1}, \ldots, \hat{y}_{L-1}\}$. (The proof is given in the appendix.)

The lemma addresses the situation in case (ii), i.e., the situation in which the source coder

output (which is also the channel input sequence) contains redundancy. Using this lemma, we can design a decoder which will take advantage of dependence in the channel input sequence. The lemma provides the mathematical structure for the decoder. The physical structure can be easily obtained by examining the quantity to be maximized. The optimum decoder maximizes $P(Y|\hat{Y})$ or equivalently $\log P(Y|\hat{Y})$, but

$$\log P(Y|\hat{Y}) = \sum \log P(y_i|\hat{Y}_i, y_{i-1}) \tag{1}$$

which is similar in form to the path metric of a convolutional decoder. Error correction using convolutional codes is made possible by explicitly limiting the possible codeword to codeword transitions, based on the previous code input and the coder structure. At the receiver the decoder compares the received data stream to the *a priori* information about the code structure. The output of the decoder is the sequence that is most likely to be the transmitted sequence. In the case where ther is residual strucure in the source coder output, while there may not be an explicit limitation on the codeword to codeword transition, the structure makes some sequences more likely to be the transmitted sequence, given a particular received sequence. In other words, while there is no structure being imposed by the encoder, there is sufficient residual structure in the source coder output that can be used for error correction. This structure can be quantified in light of the Lemma. That is, the structure is reflected in the conditional probabilities, and can be utilized via the path metric in (1) in a decoder similar in structure to a convolutional decoder. However, to implement this decoder we need to be able to compute the path metric. Unfortunately the quantity $P(y_i|\hat{Y}_i, y_{i-1})$ is difficult to estimate. We have therefore used various approximations to this quantity with some success. In [16, 17] $P(y_i|\hat{Y}_i, y_{i-1})$ is approximated by $P(y_i|\hat{y}_i, y_{i-1})$ with excellent results. Other approximations can be found in [22]. In the current work we use the

following approximation for the branch metric L.

$$L = \log \frac{P(\hat{y}_i \mid y_i) P(y_i \mid y_{i-1}, y_{i-2})}{P(\hat{y}_i)}$$

In [17] we showed that the use of the decoder led to dramatic improvements under high error rate conditions. However at low error rates the performance improvement was from nonexistent to minimal. This is in contrast to standard error correcting approaches, in which the greatest performance improvements are at low error rates, with a rapid deterioration in performance at high error rates. In this work we combine the two approaches to develop a joint source channel codec which provides protection equal to the standard channel encoders at low error rates while providing significant error protection at high error rates.

3 Convolutional Encoders and Joint Source/Channel Decoder

As convolutional coders provide excellent error protection at low error rates, and have a decoder structure similar to the JSC decoder, one way we can combine the two approaches is to obtain the transition probabilities of the convolutional encoder output and use the Joint Source/Channel (JSC) decoder described above instead of the conventional convolutional decoder. We simulated this approach using a two bit DPCM system as the source encoder. We used the three images shown in Figure 1 as the source. The USC Girl image was used for training (obtaining the requisite transition probabilities) and the Lena 256 and USC Couple images for testing. The output of the DPCM system was encoded using a (2,1,3) convolutional encoder with connection vectors

$$g^{(1)} = 64 \quad g^{(2)} = 74.$$

The convolutional encoder was obtained from [23]. The performance of the different systems was evaluated using two different measures. One was the reconstruction signal-to-noise ratio (RSNR) defined as

$$RSNR = 10 \log_{10} \frac{\sum u_i^2}{\sum (u_i - \hat{u}_i)^2}$$

where u_i is the input to the source coder (source image) and $\hat{u}_i$ is the output of the source decoder (reconstructed image). The other performance measure was the decoded error probability. The received sequence was decoded using a standard convolutional decoder and the JSC decoder. A block diagram of the system is shown in Figure 2. The results are presented in Figure 3. Notice the significant improvement in performance when the JSC decoder is used instead of the convolutional decoder. At a probability of error of 0.1 there is an improvement of about 5dB for the training set and an improvement of about 4 dB for the test set!

The simulations were repeated with a rate 2/3 (3,2,2) convolutional coder [23] with connection vectors

$$g_1^{(1)} = 7 \quad g_1^{(2)} = 1 \quad g_1^{(3)} = 4$$
$$g_2^{(1)} = 2 \quad g_2^{(2)} = 5 \quad g_2^{(2)} = 7.$$

The results are presented in Figure 4. Notice that while the rate of the code is less (2/3 as opposed to 1/2) the performance using the JSC decoder is actually better! The reason for this lies in the fact that the JSC decoder is making use of the structure in the nonbinary output of the source coder. When we used the (2,1,3) coder we destroyed some of this structure because the source coder was putting out two bit words while the channel coder was coding the input one bit at a time. However, in the case of the (3,2,2) coder, the input alphabet of the channel coder exactly matches the output alphabet of the source coder. Thus the structure in the source coder output is preserved in the channel coder output/channel input, providing better channel error protection. To verify this we conducted another set of simulation with a rate 1/2 (4,2,1) convolutional code

with connection vectors

$$g_1^{(1)} = 6 \quad g_1^{(2)} = 0 \quad g_1^{(3)} = 6 \quad g_1^4 = 4$$
$$g_2^{(1)} = 0 \quad g_2^{(2)} = 6 \quad g_2^{(3)} = 4 \quad g_2^{(4)} = 2.$$

In this case again there is a one-to-one match between the source coder output and the channel coder input, and the results shown in Figure 5 reflect this fact. There is about a two dB improvement at high error rates over the (2,1,3) rate 1/2 code, and about a one dB improvement over the rate 2/3 code. These results justify the contention that for best use of the JSC decoder the input alphabet size of the channel coder should be the same as the size of the output alphabet of the source coder. In the next section we propose a general channel coder design which is motivated by this requirement.

4 A Modified Convolutional Encoder

Given that the preservation of the structure in the source coder output requires the channel coder input alphabet to have a one-to-one match with the generally nonbinary source coder, we propose a general nonbinary convolutional encoder (NCE) whose input alphabet has the requisite property.

Let x_n, the input to the NCE, be selected from the alphabet $A = \{0, 1, 2, ..., N-1\}$, and let y_n, the output alphabet of the NCE, be selected from the alphabet $S = \{0, 1, 2, ..., M-1\}$. Then the proposed NCEs can be described by the following mappings

1. $M = N^2$; $y_n = Nx_{n-1} + x_n$

The number of bits required to represent the output alphabet using a fixed length code is

$$\lceil \log_2(M) \rceil = \lceil \log_2(N^2) \rceil = \lceil 2 \log_2(N) \rceil$$

Therefore in terms of rate, this coder is equivalent to a rate 1/2 convolutional encoder. The encoder

memory in bits is $2\lceil \log_2(N) \rceil$ as each output value depends on two input values.

As an example, consider the situation when $N = 4$. Then $A = \{0,1,2,3\}$ and $S = \{0,1,2,...,15\}$. Given the input sequence x_n : 0 1 3 0 2 1 1 0 3 3 and assuming the encoder is initialized with zeros, the output sequence will be y_n : 0 1 7 12 2 9 5 4 3 15.

The encoder memory is four bits. Notice that while the encoder output alphabet is of size N^2, at any given instant the encoder can only emit one of N different symbols as should be the case for a rate 1/2 convolutional encoder. For example if $y_{n-1} = 0$, then y_n will take on a value from $\{0,1,2,...,(N-1)\}$. In general, given a value for y_{n-1}, y_n will take on a value from $\{\alpha N, \alpha N+1, \alpha N+2, ..., \alpha N+N-1\}$, where $\alpha = y_{n-1}(mod N)$. This structure can be used by the decoder to provide error protection. The encoder is shown in Figure 1a.

2. $M = N^3$; $y_n = N^2 x_{2n-2} + N x_{2n-1} + x_{2n}$

This encoder is equivalent to a rate 1/3 convolutional encoder with an encoder memory in bits of $3\lceil \log_2(N) \rceil$. Given the same input as the previous example, the output alphabet for the NCE is

$$S = \{0,1,2,...,63\}$$

and the output sequence for the same input sequence is

$$y_n : 0\ 1\ 7\ 28\ 50\ 9\ 37\ 20\ 19\ 15$$

The encoder memory is six bits. In this case even though the encoder output alphabet is of size N^3 , at any instant the encoder can only emit one of N symbols. In general, given a value for y_{n-1}, y_n will take on a value from $\{\beta N, \beta N+1, ..., \beta N+N-1\}$, where $\beta = y_{n-1}(mod N^2)$. A block diagram of the encoder is shown in Figure 6.

3. $M = N^3$

$$y_n = N^2 x_{2n} + N x_{2n-1} + x_{2n-2}$$

The final encoder we consider is equivalent to a rate 2/3 convolutional coder. Notice that while the input output relationship looks similar to a rate 1/3 encoder, we generate one output for every two inputs. Thus, while the number of bits needed to represent one letter from the output alphabet is three times the bits needed to represent a letter from the input alphabet, because two input letters are represented by a single output letter, the rate is 2/3. Again, assuming a value of 4 for N, the output alphabet is of size 64, and for the input sequence used previously, the output sequence is y_n : 0 52 35 22 49 3.

The encoder memory is again 6 bits. The rate of the encoder can also be inferred from the fact that while the encoder output alphabet is of size N^3, at any instant the encoder can transmit one of N^2 (instead of N) symbols. Given a value for y_{n-1}, y_n can take on a value from the alphabet $\{\gamma N^2, \gamma N^2 + 1, ..., \gamma N^2 + (N^2 - 1)\}$ where $\gamma = y_{n-1}(mod N)$.

5 Binary Encoding of the NCE Output

We will make use of the residual structure in the source coder output (which is preserved in the NCE output) at the receiver. However, we can also make use of this structure in selecting binary codes for the NCE output. An intelligent assignment of binary codes can improve the error correcting performance of the system as can be seen from the following example.

Let N be 2, and let us use the rate 1/2 NCE. In this case if $y_n = 0$, y_{n+1} cannot be 2 or 3 because $y_n = 0$ means $x_n = 0$, and $y_{n+1} = 2$ or 3 means $x_n = 1$. Thus a decoded sequence cannot have 2 or 3 following 0.

Let us assign fixed length codewords to the NCE outputs as

$$0:00,\ 1:01,\ 2:10,\ 3:11$$

Now suppose the transmitted sequence was the all zero sequence, the metric used was the Hamming distance, and the received sequence is **00001000000000**; that is, there is an error in the fifth bit. If the receiver decoded the first four bits as $0,0$ then it cannot decode the fifth and sixth bits as 2 for the reason noted above. The only two options are decoding them as 0 or 1. If we decoded them as 0, we could continue decoding the rest of the sequence as $0,0\ldots$, and the Hamming distance between the received and decoded sequence would be one. If we decoded them as 1, we would have to decode the next set of two bits as 2 or 3 because 0 cannot follow 1. Decoding as 2 gives the smallest Hamming distance so we decode the seventh and eighth bit as 2. This gives a total Hamming distance of two for the incorrect path. Thus the receiver will select the correct path (the path with the smallest Hamming distance).

If the assignment had been chosen as

$$0:00;\ 1:11;\ 2:10;\ 3:01$$

then the Hamming distance for the closest incorrect path would have been three instead of two. When each allowable sequence is equally likely, there is little reason to prefer one particular assignment over others. However, when certain sequences are more likely to occur than others, it would be useful to make assignments which increase the 'distance' between likely sequences. While, for small alphabets it is a simple matter to assign the optimum binary codewords by inspection, this becomes computationally impossible for larger alphabets. We use a rather simple heuristic which, while not optimal, provides good results.

The number of M bits codewords that have to be assigned are exactly 2^M. Our strategy is therefore to try to maximize the Hamming distance between codewords that are likely to be mistaken for one another.

First we obtain a partition of the alphabet based on the fact that given a particular value for y_{n-1}, y_n can only take on values from a subset of the full alphabet. To see this, consider the rate 1/2 NCE; then the alphabet S can be partitioned into the following sub-alphabets:

$$S_0 = (0, 1, 2, 3..., N-1)$$

$$S_1 = (N, N+1, ..., 2N-1)$$

$$\vdots$$

$$S_{N-1} = \left(N(N-1), N(N-1)+1, ..., N^2-1\right)$$

where the encoder will select letters from alphabet S_j at time n if $j = y_{n-1}(\mathrm{mod} N)$. Now for each sub-alphabet we have to pick N codewords out of M $(= N^2)$ possible choices. We first pick the sub-alphabet containing the most likely letter. The letters in the sub-alphabet are ordered according to their probability of occurrence. We assign a codeword a from the list of available codewords to the most probable symbol. Then, assign the complement of a to the next symbol on the list. Therefore the distance between the two most likely symbols in the list is $K = \lceil \log_2 M \rceil$ bits. We then pick a codeword b from the list which is at a Hamming distance of $K/2$ from a and assign it and its complement to the next two elements on the list. This process is continued with the selection of letters that are$K/2^k$ away from a at the k^{th} step until all letters in the subalphabet have a codeword assigned to them. As an example, consider the case where $N = 4$. The partitions

are

$$S_0 = \{0, 1, 2, 3\}$$

$$S_1 = \{4, 5, 6, 7\}$$

$$S_3 = \{8, 9, 10, 11\}$$

$$S_4 = \{12, 13, 14, 15\}.$$

Assuming that 0 is the most probable symbol, we start by assigning codewords to the S_0 sub-alphabet. Suppose

$$P(0) \geq P(1) \geq P(2) \geq P(3).$$

We first pick a 4 bit codeword for 0 as **0000**. The next most probable symbol in this sub-alphabet is 1; therefore the codeword for 1 is the complement of the codeword for 0; 1:**1111**. The codeword for 2 is at a Hamming distance of two from the codeword for 0. The codeword 0011 satisfies this requirement; therefore the codeword for 2 is **0011** and the codeword for 3 is **1100**. Suppose the next symbol which is close in probability to the symbol 0 is 4. We select the sub-alphabet containing that symbol which is S_1. To the symbol 4 we assign a codeword from the list of unassigned codewords which is furthest from the codeword for 0. There are several possibilities for this; we pick **1011**. We then follow the same procedure for the S_1 sub-alphabet. Continuing in this manner we get the assignments shown in Table 1.

6 Simulation Results

The proposed approach was simulated using the same setup as was used in the preceding simulations. The rate 1/2 NCE and rate 2/3 NCE were simulated. The results are shown in Figures 7 and 8. Notice that the performance for the rate 1/2 NCE is about the same or slightly better than the performance of the (4,2,1) convolutional code. Similarly the performance of the rate 2/3 NCE

is about the same as the rate (3,2,2) convolutional code. Note that the memory requirements for both the NCE and the convolutional coders in both cases are the same. While the performance of the NCEs and the properly matched convolutional coders are about the same, the NCEs can be designed using a general algorithm for a source coder with any alphabet size.

7 Conclusion

If the source and channel coder are designed in a "joint" manner, that is the design of each takes into account the overall conditions (source as well as channel statistics), we can obtain excellent performance over a wide range of channel conditions. In this paper we have presented one such design. The resulting performance improvement seems to validate this approach, with a "flattening out" of the performance curves. This flattening out of the performance curves makes the approach useful for a large variety of channel error conditions.

References

[1] C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

[2] J. G. Dunham and R. M. Gray, "Joint Source and Channel Trellis Encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 516–519, July 1981.

[3] E. Ayanoğlu and R. M. Gray, "The Design of Joint Source and Channel Trellis Waveform Coders," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 855–865, November 1987.

[4] J. G. Dunham and R. M. Gray, "Joint Source and Channel Trellis Encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 516–519, July 1981.

[5] J. L. Massey, "Joint Source and Channel Coding," in *Communication Systems and Random Process Theory*, J. K. Skwirzynski, ed., Sijthoff and Nordhoff: Netherlands, pp. 279–293, 1978.

[6] T. C. Ancheta, Jr. Ph.D. dissertation, Dept. of Electrical Engr., Univ. of Notre Dame. Aug. 1977.

[7] K–Y Chang and R. W. Donaldson, "Analysis, Optimization, and Sensitivity Study of Differential PCM Systems Operating on Noisy Communication Channels," *IEEE Trans. Commun.*, vol. COM-20, pp. 338–350, June 1972.

[8] A. J. Kurtenbach and P. A. Wintz, "Quantizing for Noisy Channels," *IEEE Trans. Commun. Technol.*, vol. COM-17, pp. 291–302, April 1969.

[9] N. Farvardin and V. Vaishampayan, "Optimal Quantizer Design for Noisy Channels: An approach to Combined Source-Channel Coding," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 827–838, November 1987.

[10] D. J. Goodman and C. E. Sundberg, "Combined Source and Channel Coding for Variable Bit Rate Speech Transmission," *Bell Syst. Tech. J.*, vol. 62, pp. 2017–2036, September 1983.

[11] D. J. Goodman and C. E. Sundberg, "Transmission Errors and Forward Error Correction in Embedded Differential Pulse Code Modulation," *Bell Syst. Tech. J.*, vol. 62, pp. 2735–2764, November 1983.

[12] R. C. Reininger and J. D. Gibson, "Soft Decision Demodulation and Transform Coding of Images," *IEEE Trans. Commun.*, vol. COM-31, pp. 572–577, April 1983.

[13] R. Steele, D. J. Goodman, and C. A. McGonegal, "A Difference Detection and Correction Scheme for Combatting DPCM Transmission Errors," *IEEE Trans. Commun.*, vol. COM-27, pp. 252–255, January 1979.

[14] R. Steele, D. J. Goodman, and C. A. McGonegal, "Partial Correction of Transmission Errors in DPCM without recourse to Error Correction Coding," *Elec. Lett.*, vol. 13, pp. 351–353, June 1977.

[15] K. N. Ngan and R. Steele, "Enhancement of PCM and DPCM Images Corrupted by Transmission Errors," *IEEE Trans. Commun.*, vol. COM-30, pp. 257–269, January 1982.

[16] K. Sayood and J. C. Borkenhagen, "Utilization of Correlation in Low Rate DPCM Systems for Channel Error Protection," *Proceedings IEEE International Conference on Communications*, June 1986, pp. 1888–1892.

[17] K. Sayood and J. C. Borkenhagen, "Use of Residual Redundancy in the Design of Joint Source/Channel Coders," *IEEE Transactions on Communications*, vol. COM-39, June 1991.

[18] K. Sayood and J. D. Gibson, "Maximum Aposteriori Joint Source/Channel Coding," *Proc. 22nd Annual Conference on Information Sciences and Systems*, Princeton, NJ, Mar. 1988, pp. 380-385.

[19] J. W. Modestino, D. G. Daut, and A. L. Vickers, "Combined Source-Channel Coding of Images Using the Block Cosine Transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1262–1274, September 1981.

[20] D. Comstock and J. D. Gibson, "Hamming Coding of DCT Compressed Images Over Noisy Channels," *IEEE Trans. Commun.*, vol. COM-32, pp. 856–861, July 1984.

[21] C. C. Moore and J. D. Gibson, "Self-Orthogonal Convolutional Coding for the DPCM-AQB Speech Encoder," *IEEE Trans. Commun.*, vol. COM-32, August 1984.

[22] K. Sayood, J. D. Gibson, and F. Liu, "Implementation Issues in MAP Joint Source/Channel Coding," *Proc. 22nd Annual Asilomar Conference on Circuits, Systems, and Computers*, Nov. 1988, pp. 102-106.

[23] S. Lin and D. J. Costello, *Error Control Coding*, Prentice Hall, 1983.

Table 1: **Codeword Assignments**

Symbol	Code	Symbol	Code
0	0000	8	1011
1	0011	9	0111
2	1100	10	0100
3	1111	11	1000
4	1110	12	0101
5	1101	13	1001
6	0001	14	1010
7	0010	15	0110

Appendix 2- Item 6

An Edge Preserving Differential Image Coding Scheme*

Martin C. Rost
Sandia National Laboratories
Albuquerque, NM

Khalid Sayood
Department of Electrical Engineering and
Center for Communication and Information Science
University of Nebraska
Lincoln, NE 68588-0511

Abstract

Differential encoding techniques are fast and easy to implement. However, a major problem with the use of differential encoding for images is the rapid edge degradation encountered when using such systems. This makes differential encoding techniques of limited utility especially when coding medical or scientific images, where edge preservation is of utmost importance. We present a simple, easy to implement differential image coding system with excellent edge preservation properties. The coding system can be used over variable rate channels which makes it especially attractive for use in the packet network environment.

Introduction

The transmission and storage of digital images requires an enormous expenditure of resources, necessitating the use of compression techniques. These techniques include relatively low complexity predictive techniques such as Adaptive Differential Pulse Code Modulation (ADPCM) and its variations, as well as relatively higher complexity techniques such as transform coding and vector quantization [1,2]. Most compression schemes were originally developed for speech and their application to images is at times problematic. This is especially true of the low complexity predictive techniques. A good example of this is the highly popular ADPCM scheme. Originally designed for speech [3], it has been used with other sources with varying degrees of success. A major problem with its use in image coding is the rapid degradation in quality whenever an edge is encountered. Edges are perceptually very important and occur quite often in most images. Therefore, the degradation of edges can be perceptually very annoying. If the images under consideration contain medical or scientific data, the problem becomes even more important, as edges provide position information which may be crucial to the viewer. This poor edge reconstruction quality has been a major factor in preventing ADPCM from becoming as popular for image coding as it is for speech coding.

While good edge reconstruction capability is an important requirement for image coding schemes, another requirement that is gaining in importance with the proliferation of packet switched networks, is the ability to encode the image at different rates. In a packet switched network, the available channel capacity is not a fixed quantity, but rather fluctuates as a function of the load on the network. The compression scheme must therefore be capable of operating at different rates as the available capacity changes. This means that it should be able to take advantage of increased capacity when it becomes available while providing graceful degradation when the rate decreases to match decreased available capacity.

In this paper we describe a DPCM based coding scheme which has the desired properties listed above. It is a low complexity scheme with excellent edge preservation in the reconstructed image. It takes full advantage of the available channel capacity providing lossless compression when sufficient capacity is available, and very graceful degradation when a reduction in rate is required.

Notation and Problem Formulation

The DPCM system consists of two main blocks, the quantizer and the predictor (see Fig. 1). The predictor uses the correlation between samples of the waveform to predict the next sample value. This predicted value is removed from the waveform at the transmitter and reintroduced at the receiver. The prediction error is quantized to one of a finite number of values which is coded and transmitted to the receiver. The difference between the prediction error and the quantized prediction error is called the quantization error or the quantization noise. If the channel is error free, the reconstruction error at the receiver is simply the quantization error. To see this, note (Fig. 1) that the prediction error $e(k)$ is given by

$$e(k) = s(k) - p(k) \tag{1}$$

where the predicted value is given by

$$p(k) = \sum a_j \hat{s}(k-j) \tag{2}$$

and

*This work was supported by the NASA Goddard Space Flight Center under Grant NAG 5-916.

$$\hat{s}(k) = e_q(k) + p(k). \quad (3)$$

Assuming an additive noise model, the quantized prediction error $e_q(k)$ can be represented as

$$e_q(k) = e(k) + n_q(k) \quad (4)$$

where $n_q(k)$ denotes the quantization noise. The quantized prediction error is coded and transmitted to the receiver. If the channel is noisy this is received as $\hat{e}_q(k)$ which is given by

$$\hat{e}_q(k) = e_q(k) + n_c(k) \quad (5)$$

where $n_c(k)$ represents the channel noise. The output of the receiver $\hat{s}(k)$ is thus given by

$$\hat{s}(k) = \hat{p}(k) + \hat{e}_q(k) \quad (6)$$

where

$$\hat{p}(k) = p(k) + n_p(k) \quad (7)$$

the additional term $n_p(k)$ being the result of the introduction of channel noise into the prediction process. Using (1), (4), (5), and (7) in (6) we obtain

$$\hat{s}(k) = s(k) + n_q(k) + n_c(k) + n_p(k). \quad (8)$$

If the channel is error free, the last two terms in (8) drop out and the difference between the original and reconstructed signal is simply the quantization error.

When the prediction error is small, it falls into one of the inner levels of the quantizer, and the quantization noise is of a type referred to as granular noise. If the prediction error falls in one of the outer levels of the quantizer, the incurred quantization error is called overload noise. Because of the way the granular noise is generated it is generally smaller in magnitude than the overload noise and is bounded by the size of the quantization interval. The overload noise on the other hand is essentially unbounded and can become very large depending on the size of the prediction error. As edge pixels are rather difficult to predict, the corresponding prediction error is generally large, and this leads to large overload noise values. Furthermore, because this error effects not only the reconstruction of the current pixel, but also future predictions, the prediction errors corresponding to the next few pixels also tend to be large, leading to an edge "smearing" effect.

Reduction of the edge degradation can therefore be obtained by reducing or eliminating the slope overload noise. Reduction of the slope overload noise can be obtained by improving the prediction process. Gibson [4] analyzed ADPCM systems with backward adaptive prediction, and showed that the tracking ability of the adaptive predictor can be improved by the addition of zeros in the predictor. Motivated by these results, Sayood and Schekall [5] designed ADPCM systems for image coding with ARMA predictors. Their results show that some reduction in the edge degradation is possible with the use of adaptive zeros in the predictor. While the use of these predictors improves the edge reconstruction there is still significant degradation in the edges. One technique to further improve the edge performance was developed by Schekall and Sayood [6], which uses the Jayant quantizer as an edge detector. The overload noise is then reduced by sending a quantized representation of the noise through a side channel. The advantage of this approach is that it can be added to existing ADPCM systems. The disadvantage is that the use of a side channel introduces synchronization problems. In this paper we propose a different approach for edge preservation which does not require a side channel. This approach is described in the following section.

Proposed Approach

The approach taken in this paper is a variation on the standard rate-distortion tradeoff. The basic idea is that the slope overload noise can be reduced by increasing the rate. However rather than increasing the rate for encoding each and every pixel, there is only an instantaneous rate increase whenever slope overload is encountered. The way this is implemented is outlined in the block diagram of Figure 2. A DPCM system is followed by a lossless encoder at the transmitter. At the receiver the inverse operations are performed. The DPCM system differs from standard DPCM systems in that the quantizer being used has an unlimited number of levels. In practice what this means is that if the input has 256 levels, which is standard for monochrome images, then the DPCM quantizer will have 512 levels. This effectively eliminates the overload noise making the distortion a function of the quantizer stepsize Δ. Of course by itself it also eliminates any compression that may have been desired, in fact it requires an increase of one bit in the rate. The compression is obtained by use of the lossless encoder. The lossless encoder output alphabet consists of N codewords. These codewords correspond to N consecutive levels in the quantizer. Let the smallest level be labeled x_L and the largest level be labeled x_H. If the quantizer output $e_q(k)$ is a level between x_L and x_H, then the lossless encoder puts out the corresponding channel symbol. If, however, $e_q(k)$ is greater than x_H the encoder puts out the symbol corresponding to x_H. A new value $e_{q1}(k)$ is then obtained by subtracting x_H from $e_q(k)$. If this value is less than x_H then it is encoded using the corresponding codeword in the lossless encoder output alphabet. Otherwise, x_H is again subtracted from $e_{q1}(k)$ to generate $e_{q2}(k)$. This process is continued till some $e_{qn}(k)$ where

$$e_{qn}(k) = e_q(k) - nx_H$$

and $e_{qn}(k)$ is less than x_H. A similar strategy is followed when $e_q(k) \leq x_L$. Thus the instantaneous rate is increased by a function of n whenever the prediction error falls outside the closed interval $[x_L, x_H]$.

Example : Consider a DPCM system with a stepsize Δ of 2 where the input output relationship is given by

$$Q[x] = 2k \quad \text{if} \quad 2k - 1 \leq x < 2k + 1; \quad k = 0, \pm 1, \pm 2, \ldots$$

Let the lossless encoder output alphabet be of size eight with $x_L = -4$, and $x_H = 10$. If the input $e(k)$ is 7 the quantiger output $e_q(k)$ is 8, which is in the lossless encoder output alphabet and therefore this value is encoded as a single codeword. If $e(k)$ is 15 then $e_q(k)$ is 16, which is larger than x_H. In this case, the encoder puts out the codeword corresponding to x_H and generates $e_{q1}(k) = 16 - 10 = 6$ which is in the encoder output alphabet. Therefore, the encoder output consists of two codewords representing $x_H(10)$ and 6. If the input is $-7, e_q(k) = -6$ which is less than x_L. Thus the lossless encoder output consists of two symbols. One corresponding to the value of $x_L(-4)$ and

one corresponding to the value of -2. Note that if the input is 10 or -4 (i.e. x_H or x_L) then the output will be the sequence 10,0 or $-4,0$.

One of the consequences of this type of encoding is that it can generate runs of x_L and x_H whenever the image contains a large number of edges. Fortunately the encoding scheme also provides a significant number of special symbols that can be used to encode these runs. For example, the sequence x_H followed by a negative value and the sequence x_L followed by a positive value would not occur in the normal course of events. These sequences can therefore be used to encode the runlengths of x_L and x_H. Consider for example a system in which Δ is 2 and x_L is -4. The output of the lossless encoder therefore corresponds to the values $-4,-2,0,2,4$. In the standard system a value of 4 is always followed by a value of 0 or 2. Similarly a value of -4 is always followed by a value of 0 or -2. Therefore the sequences $4-2$ and $-4+2$ can be used as special symbols to denote runs of 4 or -4. A simple strategy is to replace every two 4's (or -4's) after the initial 4 by a -2 (or 2). For example a value of 10 would still be represented by 4 4 2. However a value of 14 would be represented by 4 -2 2 instead of 4 4 4 2. Similarly a value of 18 would be represented by 4 -2 4 2 and a value of 22 would be represented by 4 -2 -2 2. For this particular scheme, a run of length n would be represented by $n - \lfloor \frac{n-2}{2} \rfloor$ codewords. When the size of the lossless encoder output is increased, the number of special symbols available also increases and the coding of the runs can be performed more efficiently.

These special sequences can also be used to signal a change of rate for applications in which the available channel capacity changes with time. The actual change can be accomodated by changing the stepsize and reducing the lossless encoder codebook size by the same amount. Several of the systems proposed above were simulated. The results of these simulations are presented in the next section.

Results

Four systems of the type described in the previous section have been simulated. Two of the systems simulated use a one tap fixed predictor, while the other two use a one pole four zero predictor with the zeros being adaptive. One of the systems in each case contains the lossless encoder followed by a runlength encoder while the other contains only the lossless encoder without the runlength encoder. The test images used were the USC GIRL image, and the USC COUPLE image. Both are 256 X 256 monochrome eight bit images and have been used often as test images. The objective performance measure were the Peak Signal to Noise Ratio (PSNR) and the Mean Absolute Error (MAE) which are defined as follows:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{<(s(k)) - \tilde{s}(k)^2>}$$

$$\mathrm{MAE} = <|s(k) - \tilde{s}(k)|>$$

where $< \cdot >$ denotes the average value.

Several initial test runs were performed using different number of levels, different values of x_L and different values of Δ to get a feel for the optimum values of the various parameters (Given x_L and Δ, x_H is automatically determined.). We found that an appropriate way of selecting the value of x_L was using the relationship

$$x_L = -\lfloor \frac{N-1}{2} \rfloor \Delta$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x, and N is the size of the alphabet of the lossless coder. This provides a symmetric codebook when the alphabet size is odd, and a codebook skewed to the positive side when the alphabet size is even. The zero value is always in the codebook.

As the alphabet size is usually not a power of two, the binary code for the output alphabet will be a variable length code. The use of variable length codes always bring up issues of robustness with respect to changing input statistics. With this in mind, the rate was calculated in two different ways. The first was to find the output entropy, and scale it up by the ratio of symbols transmitted to the number of pixels encoded. We call this rate the entropy rate, which is the minimum rate obtainable if we assume the output of the lossless encoder to be memoryless.

While this assumption is not necessarily true, the entropy rate gives us an idea about the best we can do with a particular system. We will treat it as the lower bound on the obtainable rate. We also calculated the rate using a predetermined variable length code. This code was designed with no prior knowledge of the probabilities of the different letters. The only assumption was that the letters representing the inner levels of the quantizer were always more likely than the letters representing the outer levels of the quantizer. The code tree used is shown in Figure 3. Obviously, this will become highly inefficient in the case of small alphabet size and small Δ, as in this case, the outer levels x_L and x_H will occur quite frequently. This rate can be viewed as an upper bound on the achievable rate.

The results for the system with a one tap predictor and without the runlength encoder are shown in Tables 1 and 2. Table 1 contains the results for the COUPLE image, while Table 2 contains the results for the GIRL image. In the table R_L denotes the entropy rate while R_U is the rate obtained using the Huffman code of Figure 3. Recall that for image compression schemes, systems with PSNR values of greater than 35 dB are perceptually almost identical. As can be seen from the PSNR values in the tables there is very little degradation with rate, and in fact if we use the 35 dB criterion there is almost no degradation in image quality until the rate drops below two bits per pixel. This can be verified by the reconstructed images shown in Figure 4. Each picture in Figure 4 consists of the original image, the reconstructed image and the error image magnified 10 fold. In each of the pictures, it is extremely difficult to tell the source or original image from the reconstructed or output image. In fact, in the case of the image coded at rates above two bits per pixel it is well nigh impossible. This subjective observation is supported by the error images in each case which are uniform in texture throughout without any of the standard edge artifacts which can be usually seen in the error images for most compression schemes.

We can see from the results that if the value of Δ and hence x_L is fixed, the size of the codebook has no effect in on the performance measures. This is because the only effect of reducing the codebook size under these conditions is to increase the number of symbols transmitted. While this has the effect of increasing the rate, because of the way the system is constructed, it does not influence the resulting distortion. The drop in rate for the same distortion as the alphabet size increases can be clearly seen from the results in Tables 1 and 2.

Table 3 shows the decrease in rate when a simple runlength coder is used. The runlength coder encodes long strings of x_L and x_H using the special sequences mentioned previously. As can be seen from the results the improvement provided by the current runlength encoding scheme is significant only for small alphabets and small values of Δ. This is because it is under these conditions that most of the long strings of x_L and x_H are generated. However we are not as yet using many of the special sequences in the larger alphabet codebooks, so there is certainly room for improvement.

The one tap predictor was replaced with an adaptive ARMA predictor with a fixed pole and four adaptive zeros. The fixed pole was at a lag of 257 (pixel above) while the zeros were at lags of one, two, three and four. The adaption was performed using a sample LMS algorithm as follows. Let $\mathbf{B}_k$ be the vector of predictor coefficients at time k. The adaption algorithm was

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mu \epsilon_q(k) \mathbf{E}_k$$

where μ is the adaption stepsize and

$$\mathbf{E}_k = (\epsilon_q(k-1), \epsilon_q(k-2), \epsilon_q(k-3), \epsilon_q(k-4))^T .$$

The results from using this predictor are shown in Tables 4, 5 and 6. While there is some improvement in all cases, the results for the COUPLE image show a greater improvement than the results for the GIRL image. This can be explained by noting that the COUPLE image contains many more edges than the GIRL image. As the ARMA predictor tends to improve predictor performance when edges are encountered, the improvement in performance occurs in the image with more edges.

Conclusion

We have demonstrated a simple image coding scheme which is very easy to implement in realtime and has excellent edge preservation properties over a wide range of rates.

This system would be especially useful in transmitting images over channels where the available bandwidth may be vary. The edge preserving quality is especially useful in the encoding of scientific and medical images.

References

[1] N. S. Jayant and P. Noll, *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[2] A. K. Jain, "Image Data Compression: A Review," *Proc. IEEE*, vol. 69, pp. 349-389, Mar. 1981.

[3] C. C. Cutler, "Differential Quantization for Communication Signals," U.S. Patent 2 605 361, July 29, 1952.

[4] J. D. Gibson, "Backward Adaptive Prediction as Spectral Analysis Within a Closed Loop," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1166-1174, Oct. 1985.

[5] K. Sayood and S. M. Schekall, "Use of ARMA Predictors in the Differential Encoding of Images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-36, pp. 1791-1795, Nov. 1988.

[6] S. M. Schekall and K. Sayood, "An Edge Preserving DPCM Scheme for Image Coding," *Proc. 31st Midwest Symposium on Circuits and Systems*, St. Louis, pp. 901-907, Aug. 1988.

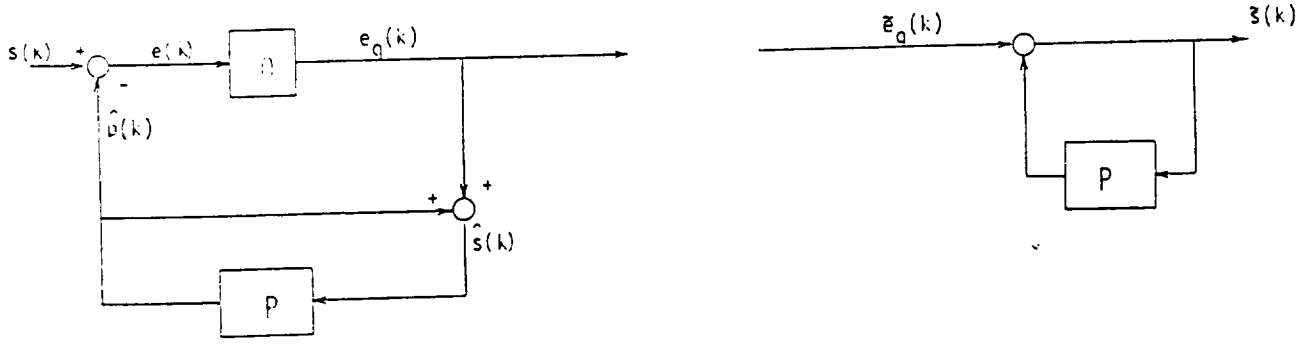


Figure 1. Block Diagram of a DPCM System

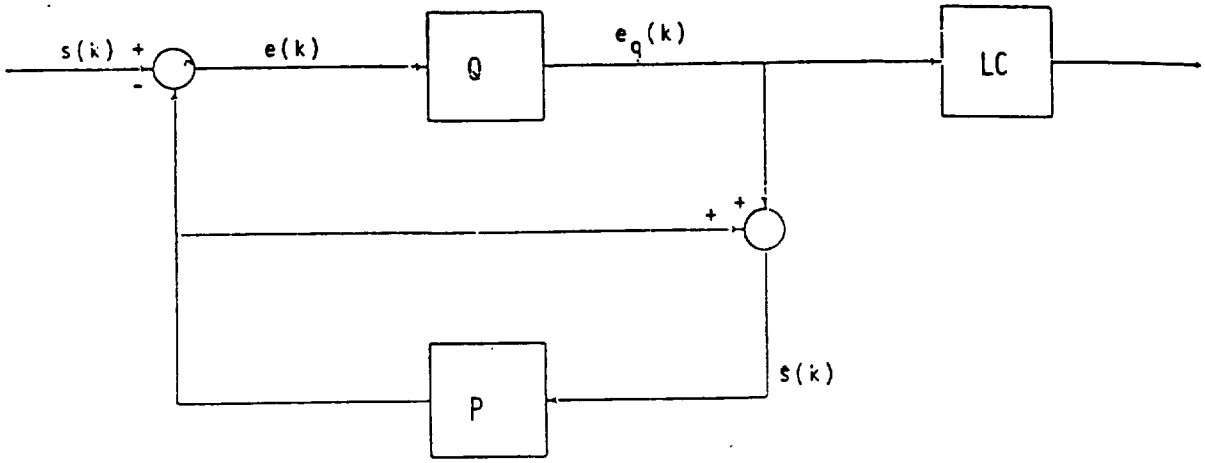


Fig. 2 Proposed Encoder Structure

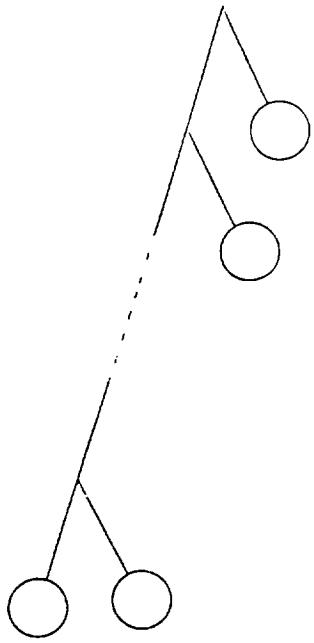


Fig. 3 Variable Length Code Tree

Delta	MAE	PSNR (dB)	Size = 3 R_L	Size = 3 R_U	Size = 5 R_L	Size = 5 R_U	Size = 8 R_L	Size = 8 R_U
2	0.5067	51.0830	6.1615	7.1418	4.9334	6.8635	4.4404	6.6884
4	1.4790	42.7898	3.8909	4.0587	3.3637	2.7982	3.1673	3.6939
6	2.4676	38.6565	2.9577	3.0137	2.6553	2.7729	2.5490	2.7023
8	3.3697	36.0009	2.4314	2.4972	2.2327	2.2756	2.1662	2.2267
12	5.1359	32.3682	1.8277	1.9800	1.7233	1.7963	1.6930	1.7669

Table 1: Performance results for the COUPLE image, alphabet size 3, 5 and 8.

Delta	MAE	PSNR (dB)	Size = 3 R_L	Size = 3 R_U	Size = 5 R_L	Size = 5 R_U	Size = 8 R_L	Size = 8 R_U
2	0.5067	51.0830	6.2821	7.8120	5.0554	7.4713	4.5635	7.1275
4	1.4790	42.7898	4.0088	4.3976	3.7414	4.0592	3.2668	3.8740
6	2.4676	38.6565	3.0819	3.2547	2.7570	2.9279	2.6468	2.8063
8	3.3697	36.0009	2.5543	2.6860	2.3272	2.3783	2.2617	2.2931
12	5.1359	32.3682	1.9426	2.1122	1.8046	1.8439	1.7786	1.8009

Table 2: Performance results for the GIRL image, alphabet size 3, 5 and 8.

Delta	Size = 3 Without RL Encoder	Size = 3 With RL Encoder	Size = 5 Without RL Encoder	Size = 5 With RL Encoder	Size = 8 Without RL Encoder	Size = 8 With RL Encoder
2	6.16	5.44	4.93	4.34	4.44	4.29
4	3.89	3.60	3.36	3.25	3.16	3.15
6	2.96	2.81	2.66	2.63	2.55	2.55
8	2.43	2.35	2.23	2.22	2.17	2.17
12	1.83	1.80	1.72	1.72	1.69	1.69

Table 3: Comparison of Entropy rates between system with Runlength (RL) Encoder and without RL Encoder for COUPLE image.

Delta	MAE	PSNR (dB)	Size = 3 R_L	Size = 3 R_U	Size = 5 R_L	Size = 5 R_U	Size = 8 R_L	Size = 8 R_U
2	1.59	46.11	4.71	5.00	3.94	4.77	3.63	4.69
4	2.00	40.71	3.02	3.04	2.70	2.82	2.50	2.76
6	2.96	37.42	2.33	2.38	2.14	2.18	2.09	2.13
8	3.86	35.11	1.94	2.05	1.81	1.87	1.79	1.83
12	5.61	31.79	1.49	1.72	1.42	1.56	1.41	1.55

Table 4: Performance results for COUPLE image with adaptive ARMA predictor.

Delta	MAE	PSNR (dB)	Size = 3 R_L	Size = 3 R_U	Size = 5 R_L	Size = 5 R_U	Size = 8 R_L	Size = 8 R_U
2	1.07	45.99	5.66	6.33	4.59	6.06	4.18	5.92
4	2.06	40.55	3.60	3.69	3.15	3.42	2.99	3.32
6	3.06	37.15	2.78	2.82	2.51	2.56	2.42	2.48
8	4.04	34.75	2.31	2.38	2.12	2.14	2.07	2.09
12	6.08	31.23	1.79	1.95	1.66	1.73	1.65	1.70

Table 5: Performance results for GIRL image with adaptive ARMA predictor.

	Size = 3		Size = 5		Size = 8	
Delta	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder
2	4.71	4.25	3.94	3.70	3.63	3.57
4	3.02	2.86	2.70	2.67	2.60	2.59
6	2.33	2.26	2.14	2.13	2.09	2.09
8	1.94	1.90	1.81	1.81	1.79	1.79
12	1.49	1.48	1.42	1.42	1.41	1.41

Table 6: Comparison of Entropy rates between systems with and without the Runlength Encoder for the COUPLE image.

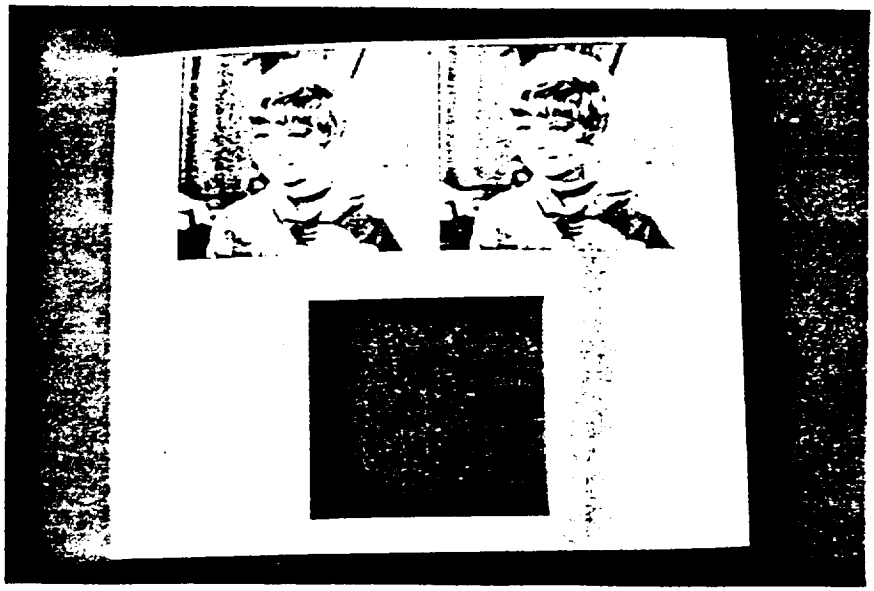


Figure 4(a). GIRL image coded at entropy rate of 1.7 bpp.

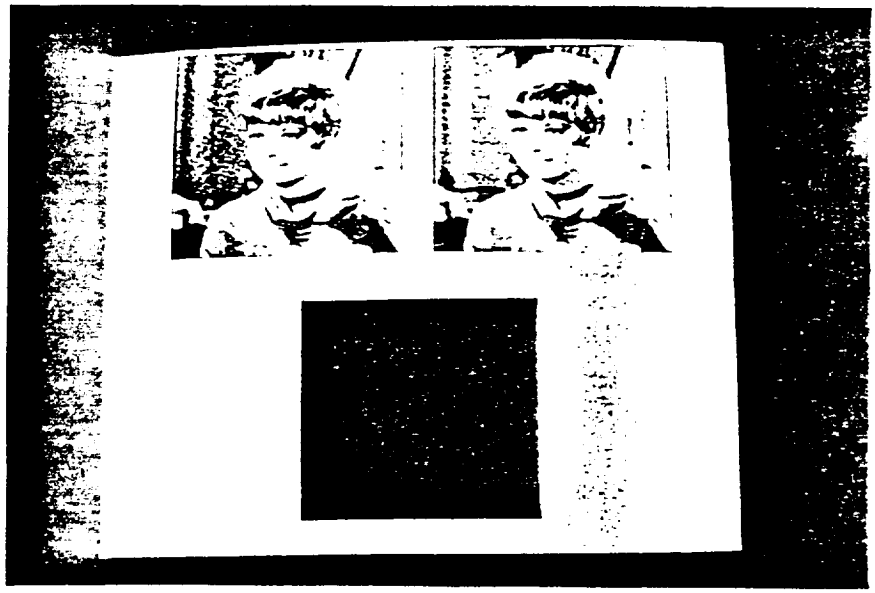


Figure 4(b). GIRL image coded at entropy rate of 1.5 bpp.

Appendix 2- Item 7

An Edge Preserving Differential Image Coding Scheme[1]

Martin C. Rost
Sandia National Laboratories
Albuquerque, NM

and

Khalid Sayood
Department of Electrical Engineering
and
Center for Communication and Information Science
University of Nebraska
Lincoln, NE 68588-0511

EDICS Category : 6.3.2

[1]This work was supported by the NASA Goddard Space Flight Center under Grant NAG 5-916.

An Edge Preserving Differential Image Coding Scheme

Abstract

Differential encoding techniques are fast and easy to implement. However, a major problem with the use of differential encoding for images is the rapid edge degradation encountered when using such systems. This makes differential encoding techniques of limited utility especially when coding medical or scientific images, where edge preservation is of utmost importance. We present a simple, easy to implement differential image coding system with excellent edge preservation properties. The coding system can be used over variable rate channels which makes it especially attractive for use in the packet network environment.

1. Introduction

The transmission and storage of digital images requires an enormous expenditure of resources, necessitating the use of compression techniques. These techniques include relatively low complexity predictive techniques such as Adaptive Differential Pulse Code Modulation (ADPCM) and its variations, as well as relatively higher complexity techniques such as transform coding and vector quantization [1,2]. Most compression schemes were originally developed for speech and their application to images is at times problematic. This is especially true of the low complexity predictive techniques. A good example of this is the highly popular ADPCM scheme. Originally designed for speech [3], it has been used with other sources with varying degrees of success. A major problem with its use in image coding is the rapid degradation in quality whenever an edge is encountered. Edges are perceptually very important and therefore their degradation can be perceptually very annoying. If the images under consideration contain medical or scientific data, the problem becomes even more important, as edges provide position information which may be crucial to the viewer. This poor edge reconstruction quality has been a major factor in preventing ADPCM from becoming as popular for image coding as it is for speech coding. While good edge reconstruction capability is an important requirement for image coding schemes, another requirement that is gaining in importance with the proliferation of packet switched networks, is the ability to encode the image at different rates. In a packet switched network, the available channel capacity is not a fixed quantity, but rather fluctuates as a function of the load on the network. The compression scheme must therefore be capable of taking advantage of increased capacity when it becomes available while providing graceful degradation when the rate decreases to match decreased available capacity.

In this paper we describe a DPCM based coding scheme which has the desired properties listed above. It is a low complexity scheme with excellent edge preservation in the reconstructed image. It

takes full advantage of the available channel capacity providing lossless compression when sufficient capacity is available, and very graceful degradation when a reduction in rate is required.

2. Notation and Problem Formulation

The DPCM system consists of two main blocks, the quantizer and the predictor (see Fig. 1). The predictor uses the correlation between samples of the waveform $s(k)$ to predict the next sample value. This predicted value is removed from the waveform at the transmitter and reintroduced at the receiver. The prediction error is quantized to one of a finite number of values which is coded and transmitted to the receiver and is denoted by $e_q(k)$. The difference between the prediction error and the quantized prediction error is called the quantization error or the quantization noise. If the channel is error free, the reconstruction error at the receiver is simply the quantization error. To see this, note (Fig. 1) that the prediction error $e(k)$ is given by

$$e(k) = s(k) - p(k) \tag{1}$$

where $s(k)$ is original signal predicted by $p(k)$ which is given by

$$p(k) = \sum a_j \hat{s}(k-j), \tag{2}$$

$$\hat{s}(k) = e_q(k) + p(k). \tag{3}$$

Assuming an additive noise model, the quantized prediction error $e_q(k)$ can be represented as

$$e_q(k) = e(k) + n_q(k) \tag{4}$$

where $n_q(k)$ denotes the quantization noise. The quantized prediction error is coded and transmitted to the receiver. If the channel is noisy this is received as $\tilde{e}_q(k)$ which is given by

$$\tilde{e}_q(k) = e_q(k) + n_c(k) \tag{5}$$

where $n_c(k)$ represents the channel noise. The output of the receiver $\tilde{s}(k)$ is thus given by

$$\tilde{s}(k) = \tilde{p}(k) + \tilde{e}_q(k), \tag{6}$$

$$\tilde{p}(k) = p(k) + n_p(k) \tag{7}$$

the additional term $n_p(k)$ being the result of the introduction of channel noise into the prediction process. Using (1), (4), (5), and (7) in (6) we obtain

$$\tilde{s}(k) = s(k) + n_q(k) + n_c(k) + n_p(k). \tag{8}$$

If the channel is error free, the last two terms in (8) drop out and the difference between the original and reconstructed signal is simply the quantization error.

When the prediction error is small, it falls into one of the inner levels of the quantizer, and the quantization noise is of a type referred to as granular noise. If the prediction error falls in one of the outer levels of the quantizer, the incurred quantization error is called overload noise. Granular noise is generally smaller in magnitude than the overload noise and is bounded by the size of the quantization interval. The overload noise on the other hand is essentially unbounded and can become very large depending on the size of the prediction error. As edge pixels are rather difficult to predict, the corresponding prediction error is generally large, and this leads to large overload noise values. Furthermore, because this error effects not only the reconstruction of the current pixel, but also future predictions, the prediction errors corresponding to the next few pixels also tend to be large, leading to an edge "smearing" effect.

Reduction of the edge degradation can therefore be obtained by reducing or eliminating the slope overload noise. Reduction of the slope overload noise can be obtained by improving the prediction process. Gibson [4] analyzed ADPCM systems with backward adaptive prediction, and showed that the tracking ability of the adaptive predictor can be improved by the addition of zeros in the predictor. Motivated by these results, Sayood and Schekall [5] designed ADPCM systems for image coding with ARMA predictors. Their results show that some reduction in the edge degradation is possible with the use of adaptive zeros in the predictor. While the use of these predictors improves the edge reconstruction there is still significant degradation in the edges. One technique to further improve the edge performance was developed by Schekall and Sayood [6], which uses the Jayant quantizer as an edge detector. The overload noise is then reduced by sending a quantized representation of the noise through a side channel. The advantage of this approach is that it can be added to existing ADPCM systems. The disadvantage is that the use of a side channel introduces synchronization problems. In this paper we propose a different approach for

edge preservation which does not require a side channel. This approach is described in the following section.

3. Proposed Approach

The approach taken in this paper is a variation on the standard rate-distortion tradeoff. The basic idea is that the slope overload noise can be reduced by increasing the rate. However rather than increasing the rate for encoding each and every pixel, there is only an instantaneous rate increase whenever slope overload is encountered. The way this is implemented is outlined in the block diagram of Fig. 2. A DPCM system is followed by a lossless encoder at the transmitter. At the receiver the inverse operations are performed. The DPCM system differs from standard DPCM systems in that the quantizer being used effectively has an unlimited number of levels. In practice what this means is that if the input has 256 levels, which is standard for monochrome images, then the DPCM quantizer will have 512 levels. This effectively eliminates the overload noise making the distortion a function of the quantizer stepsize Δ. Of course by itself it also eliminates any compression that may have been desired, in fact it requires an increase of one bit in the rate. The compression is obtained by use of the lossless encoder. The lossless encoder output alphabet consists of N codewords. These codewords correspond to N consecutive levels in the quantizer. Let the smallest level be labeled x_L and the largest level be labeled x_H. If the quantizer output $e_q(k)$ is a level between x_L and x_H, then the lossless encoder puts out the corresponding channel symbol. If, however, $e_q(k)$ is greater than x_H, the encoder puts out n symbols corresponding to x_H and a symbol corresponding to $e_{qn}(k)$ where

$$n = \lfloor e_q(k)/_{x_H} \rfloor \text{ and } e_{qn}(k) = e_q(k) \pmod{x_H}$$

A similar strategy is followed when $e_q(k) \leq x_L$. Thus the instantaneous rate is increased by a function of n whenever the prediction error falls outside the closed interval $[x_L, x_H]$.

One of the consequences of this type of encoding is that it can generate runs of x_L and x_H whenever the image contains a large number of edges. Fortunately the encoding scheme also provides a significant number of special symbols (x_H followed by a symbol for a negative level and x_L followed by a symbol for a positive level) that can be used to encode these runs. When the size of the lossless encoder output is increased, the number of special symbols available also increases and the coding of the runs can be performed more efficiently.

These special sequences can also be used to signal a change of rate for applications in which

the available channel capacity changes with time. The actual change can be accommodated by changing the stepsize and reducing the lossless encoder codebook size by the same amount. Several of the systems proposed above were simulated. The results of these simulations are presented in the next section.

4. Results

Before we provide the results using images, let us examine the performance of the scheme when applied to a one-dimensional signal containing a simulated edge. This signal was first encoded using a five-level quantizer. The results are shown in Fig. 4(a). As can be seen, it takes a little while for the DPCM system to catch up. In an image this would cause a smearing of the edge. When the proposed system with the same parameters is used there is no such effect, as is clear from Fig. 4(b). The quantizer in this case went into the recursive mode twice, once at the leading and once at the trailing edge. To get an equivalent effect, a standard DPCM system would have to have a forty-level quantizer. To show that this performance is maintained when the system is used with 2D images, two systems of the type described in the previous section have been simulated. Both systems use the following two-dimensional fixed predictor [7]: $p(k) = 2/3\,\hat{s}(k-1) + 2/3\,\hat{s}(k-256) - 1/3\,\hat{s}(k-257)$. One of the systems contains the lossless encoder followed by a runlength encoder while the other contains only the lossless encoder without the runlength encoder. The test images used were the USC GIRL image, and the USC COUPLE image. Both are 256 X 256 monochrome eight bit images and have been used often as test images. The objective performance measure were the Peak Signal to Noise Ratio (PSNR) and the Mean Absolute Error (MAE) which are defined as follows:

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2}{<(s(k)) - \tilde{s}(k)^2>}$$

$$\mathrm{MAE} = <|s(k) - \tilde{s}(k)|>$$

where $< \cdot >$ denotes the average value.

Several initial test runs were performed using different number of levels, different values of x_L and different values of Δ to get a feel for the optimum values of the various parameters (Given x_L and Δ, x_H is automatically determined.). We found that an appropriate way of selecting the value of x_L was using the relationship

$$x_L = -\lfloor\frac{N-1}{2}\rfloor\Delta$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x, and N is the size of the alphabet of the lossless coder. This provides a symmetric codebook when the alphabet size is odd, and a codebook skewed to the positive side when the alphabet size is even. The zero value is always in the codebook.

As the alphabet size is usually not a power of two, the binary code for the output alphabet will be a variable length code. The use of variable length codes always bring up issues of robustness with respect to changing input statistics. With this in mind, the rate was calculated in two different ways. The first was to find the output entropy, and scale it up by the ratio of symbols transmitted to the number of pixels encoded. We call this rate the entropy rate, which is the minimum rate obtainable if we assume the output of the lossless encoder to be memoryless. While this assumption is not necessarily true, the entropy rate gives us an idea about the best we can do with a particular system. We also calculated the rate using a predetermined variable length code. This code was designed with no prior knowledge of the probabilities of the different letters. The only assumption was that the letters representing the inner levels of the quantizer were always more likely than the letters representing the outer levels of the quantizer. The code tree used is shown in Fig. 3. Obviously, this will become highly inefficient in the case of small alphabet size and small Δ, as in this case, the outer levels x_L and x_H will occur quite frequently. This rate can be viewed as an upper bound on the achievable rate.

The results for the system without the runlength encoder are shown in Tables 1 and 2. Table 1 contains the results for the COUPLE image, while Table 2 contains the results for the GIRL image. In the table R_L denotes the entropy rate while R_U is the rate obtained using the Huffman code of Fig. 3. Recall that for image compression schemes, systems with PSNR values of greater than 35 dB are perceptually almost identical. As can be seen from the PSNR values in the tables there is very little degradation with rate, and in fact if we use the 35 dB criterion there is almost no degradation in image quality until the rate drops below two bits per pixel. This can be verified by the reconstructed images shown in Fig. 5. Each picture in Fig. 5 consists of the original image, the reconstructed image and the error image magnified 10 fold. In each of the pictures, it is extremely difficult to tell the source or original image from the reconstructed or output image. This subjective observation is supported by the error images in each case which are uniform in texture throughout without the edge artifacts which can be usually seen in the error images for most compression schemes.

We can see from the results that if the value of Δ and hence x_L is fixed, the size of the codebook has no effect on the performance measures. This is because the only effect of reducing the codebook

size under these conditions is to increase the number of symbols transmitted. While this has the effect of increasing the rate, because of the way the system is constructed it does not influence the resulting distortion. The drop in rate for the same distortion as the alphabet size increases can be clearly seen from the results in Tables 1 and 2.

Table 3 shows the decrease in rate when a simple runlength coder is used. The runlength coder encodes long strings of x_L and x_H using the special sequences mentioned previously. As can be seen from the results the improvement provided by the current runlength encoding scheme is significant only for small alphabets and small values of Δ. This is because it is under these conditions that most of the long strings of x_L and x_H are generated. However we are not as yet using many of the special sequences in the larger alphabet codebooks, so there is certainly room for improvement.

Finally to show the effect of changing rate on the perceptual quality, the USC GIRL image was encoded using three different rates. The top quarter of the image was encoded using a codebook size of eight and a Δ of two resulting in a rate of 4.37. The second quarter of the image was encoded using a codebook of size five and a Δ of 4 resulting in a rate of 2.86. The bottom half of the image was encoded using a codebook size of three and Δ of eight resulting in a rate of 2.36. The original and reconstructed images are shown in Fig. 6. The fact that the image is coded with three different rates can only be noticed if the viewer is already aware of this fact and then only after very close scrutiny. The fact that the image was encoded using three different rates is clear though in the magnified error image shown in Fig. 7. This property of the coding scheme would be extremely useful if changes in the transmission bandwidth forced the coder to operate at different rates.

To see how this algorithm performs on a relative scale, we compare it to the differential scheme proposed by Maragos, Shafer and Mersereau [8]. The system proposed by Maragos et. al. uses a forward adaptive two dimensional predictor and a backward adaptive quantizer. The coefficients are obtained over a 32x32 or a 16x16 frame and transmitted as side information. The proposed system (we feel) is considerably simpler, because of the lack of any need for adaptation and side information; however, the results compare favorably with the system of [8]. Comparative results are shown in Table 5. The results were obtained by varying the stepsize Δ until the rate obtained was similar to the rate in [8], and then comparing the PSNR. As in [8], to obtain rates below one bits/pixel several coder outputs were concatenated into blocks which were then Huffman encoded. For the results shown in Table 5, we used a block size of three. Given a five-level recursive quantizer, this corresponds to an alphabet size of 125, which would be somewhat excessive for a simple

implementation. (In [8] block sizes of four to eight are used with two- and three-level quantizers.)

The above comparison is not meant to indicate that the two systems being compared are exclusive. A case can be made for combining the good features of both systems. For example, the prediction scheme described in [8] could be combined with the quantization scheme described here. However, it was felt in this particular case that the advantages to be gained by the addition of a forward adaptive predictor were offset by the increase in complexity and synchronization requirements.

5. Conclusion

We have demonstrated a simple image coding scheme which is very easy to implement in realtime and has excellent edge preservation properties over a wide range of rates.

This system would be especially useful in transmitting images over channels where the available bandwidth may be vary. The edge preserving quality is especially useful in the encoding of scientific and medical images.

6. References

[1] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[2] A. K. Jain, "Image Data Compression: A Review," *Proc. IEEE*, vol. 69, pp. 349–389, Mar. 1981.

[3] C. C. Cutler, "Differential Quantization for Communication Signals," U.S. Patent 2 605 361, July 29, 1952.

[4] J. D. Gibson, "Backward Adaptive Prediction as Spectral Analysis Within a Closed Loop," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1166–1174, Oct. 1985.

[5] K. Sayood and S. M. Schekall, "Use of ARMA Predictors in the Differential Encoding of Images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-36, pp. 1791–1795, Nov. 1988.

[6] S. M. Schekall and K. Sayood, "An Edge Preserving DPCM Scheme for Image Coding," *Proc. 31st Midwest Symposium on Circuits and Systems*, St. Louis, pp. 904–907, Aug. 1988.

[7] C. W. Harrison, "Experiments with Linear Prediction in Television," *Bell Syst. Tech. J.*, vol. 31, pp. 764-783, July 1952.

[8] P. A. Maragos, R. W. Schafer, and R. M. Mersereau, "Two-Dimensional Linear Prediction and Its Application to Adaptive Predictive Coding of Images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-32, pp. 1213-1229, Dec. 1984.

Delta	MAE	PSNR	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	0.50	51.12	4.45	4.89	3.88	4.79	3.66	4.79
4	0.96	46.56	2.82	2.85	2.64	2.77	2.58	2.76
8	1.86	41.17	1.78	1.85	1.74	1.80	1.73	1.79
16	3.54	35.63	1.06	1.37	1.05	1.35	1.05	1.34
32	6.82	30.04	0.56	1.14	0.55	1.13	0.55	1.13

Table 1: Performance results for the COUPLE image, alphabet size 3, 5 and 8.

Delta	MAE	PSNR	Size = 3		Size = 5		Size = 8	
			R_L	R_U	R_L	R_U	R_L	R_U
2	0.49	51.17	5.01	5.78	4.26	5.66	4.01	5.61
4	0.99	46.37	3.17	3.28	2.93	3.17	2.86	3.14
8	1.98	40.79	2.02	2.05	1.96	1.99	1.95	1.97
16	3.74	35.24	1.21	1.44	1.19	1.42	1.19	1.41
32	6.90	29.86	0.63	1.16	0.63	1.16	0.63	1.16

Table 2: Performance results for the GIRL image, alphabet size 3, 5 and 8.

	Size = 3		Size = 5		Size = 8	
Delta	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder
2	5.01	4.58	4.26	4.06	4.01	3.95
4	3.17	3.05	2.93	2.90	2.86	2.86
8	2.02	2.01	1.96	1.96	1.95	1.95
16	1.21	1.21	1.19	1.19	1.19	1.19
32	0.63	0.63	0.63	0.63	0.63	0.63

Table 3: Comparison of Entropy rates between systems with Runlength (RL) Encoder and without RL Encoder for GIRL image.

	Size = 3		Size = 5		Size = 8	
Delta	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder	Without RL Encoder	With RL Encoder
2	4.45	4.05	3.88	3.66	3.66	3.61
4	2.82	2.69	2.64	2.61	2.58	2.58
8	1.78	1.75	1.74	1.73	1.73	1.73
16	1.06	1.06	1.05	1.05	1.05	1.05
32	0.56	0.55	0.55	0.55	0.55	0.55

Table 4: Comparison of Entropy rates between systems with and without the Runlength Encoder for the COUPLE image.

Results from [8] (Frame size=32, 3 level AQB)		Results from proposed system (alphabet size 5)	
Rate	PSNR	Rate	PSNR
0.74	30.3	0.74	31.13
0.83	31.6	0.84	32.1
0.93	32.6	0.94	33.1
1.03	33.4	1.03	33.9

Table 5: Comparison of proposed system with that of [8].

List of Figures

Figure 1: Block Diagram of a DPCM System

Figure 2: Proposed Encoder Structure

Figure 3: Variable Length Code Tree

Figure 4: Coding of Simulated 1-D edge with (a) DPCM, (b) Proposed system

Figure 5(a): GIRL Image Coded at Entropy Rate of 1.5 bpp

Figure 5(b): GIRL Image Coded at Entropy Rate of 1.3 bpp

Figure 6: GIRL Image Coded at Three Different Rates

Figure 7: Error Image for GIRL Image Coded at Three Different Rates.

List of Tables

Table 1: Performance results for the COUPLE image, alphabet size 3, 5 and 8.

Table 2: Performance results for the GIRL image, alphabet size 3, 5 and 8.

Table 3: Comparison of Entropy rates between system with Runlength (RL) Encoder and without RL Encoder for GIRL image.

Table 4: Comparison of Entropy rates between systems with and without the Runlength Encoder for the COUPLE image.

Table 5: Comparison of proposed system with that of [8].

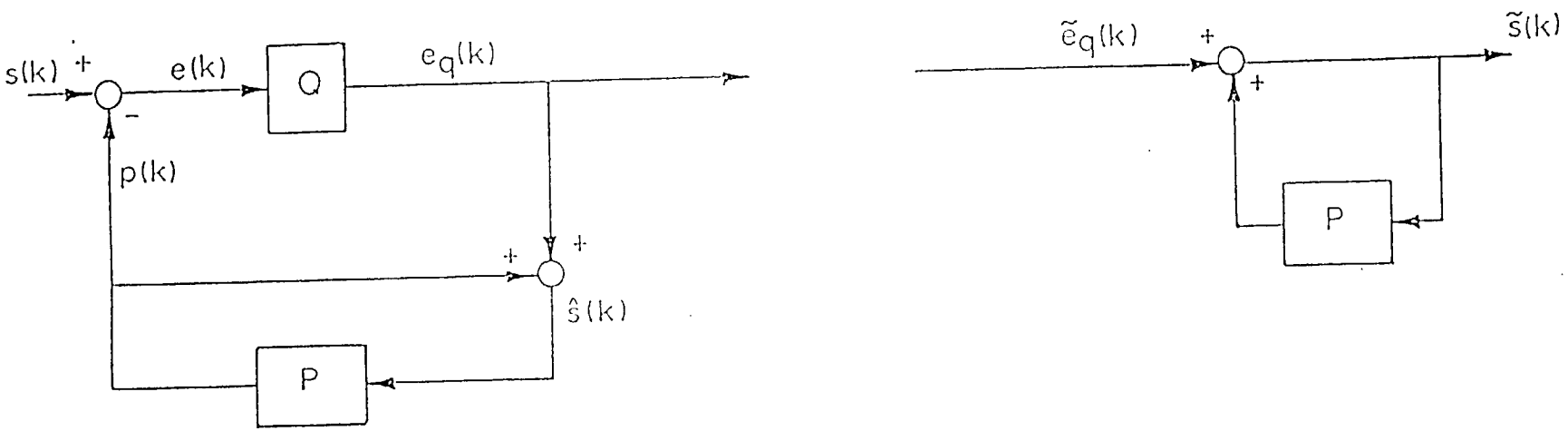


Figure 1. Block Diagram of DPCM System

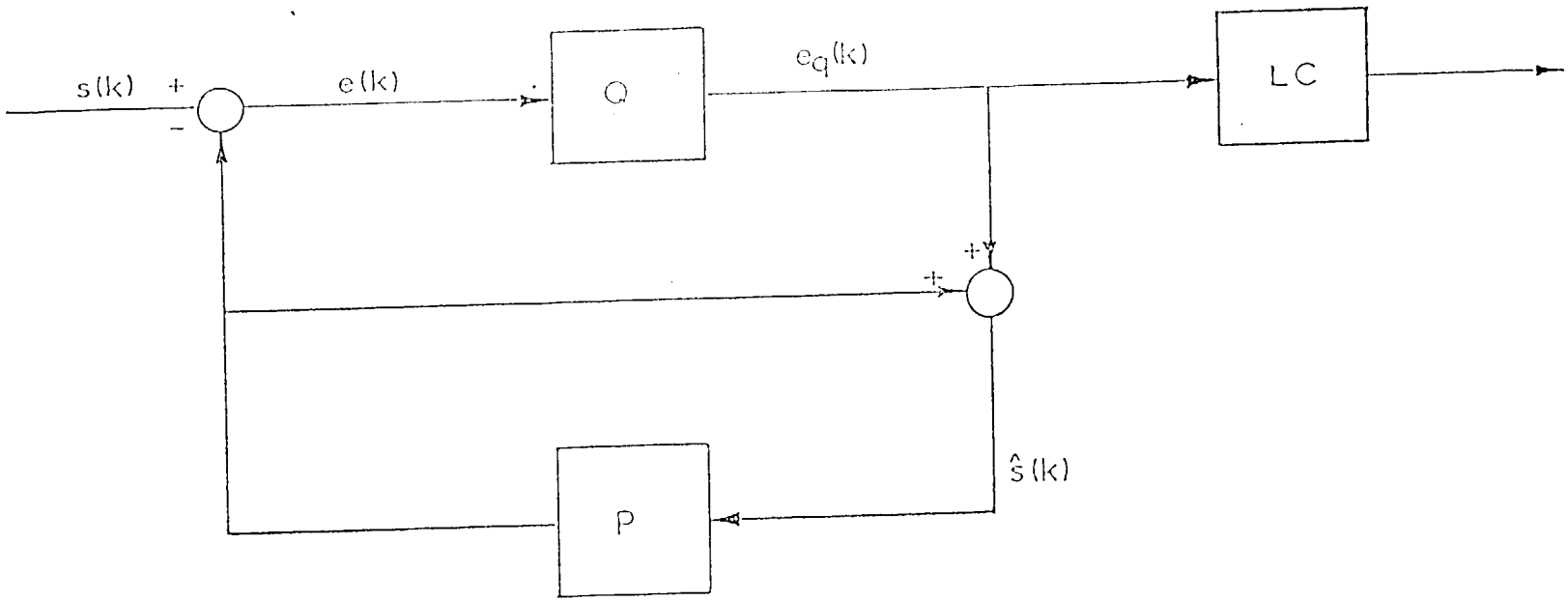


Figure 2. Proposed Encoder Structure

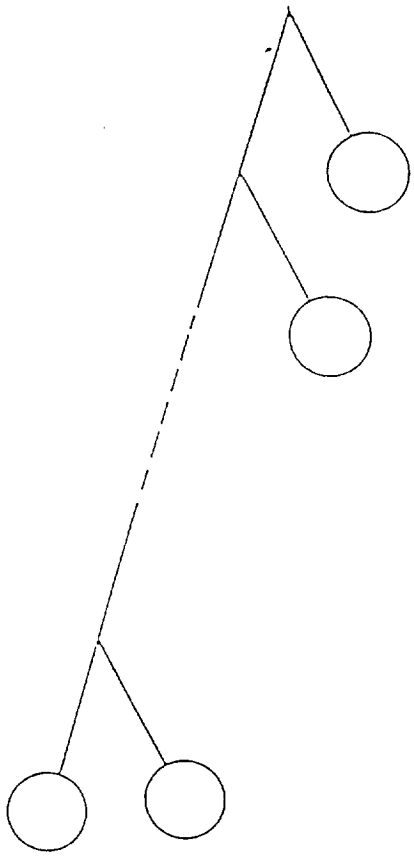


Figure 3. Variable Length Code Tree

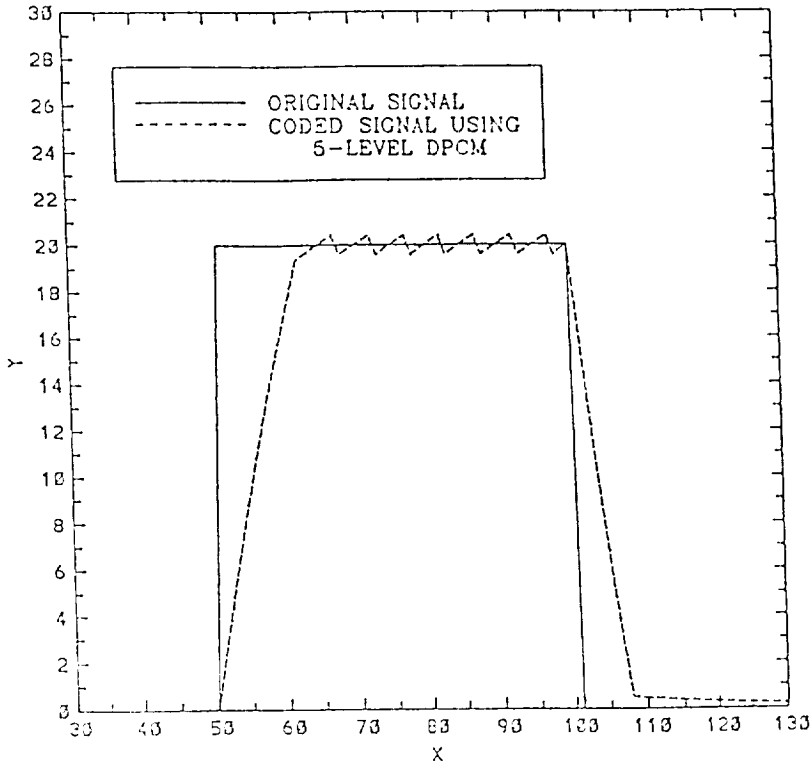


Figure 4(a) Simulated edge encoded using 5-level DPCM

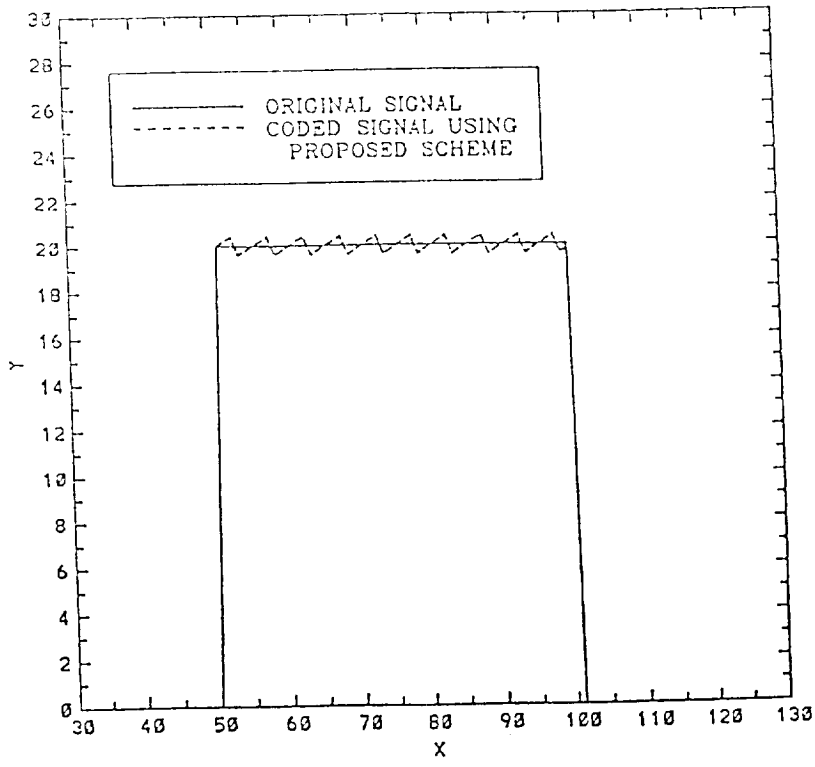
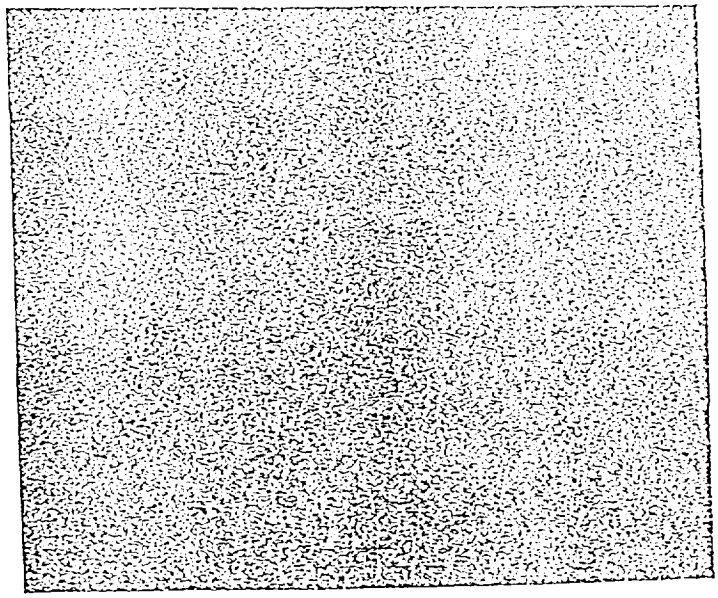


Figure 4(b) Simulated edge encoded using proposed scheme

Original Image

Coded Image

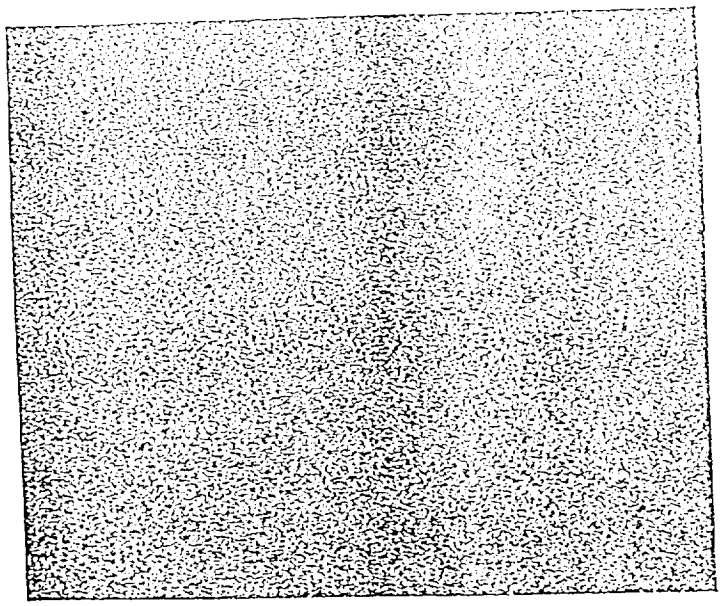


Error Image

Figure 5(a) GIRL Image Coded at entropy rate of 1.5 bpp

Original Image

Coded Image



Error Image

Figure 5(b) GIRL Image Coded at entropy rate of 1.3 bpp

Original Image

Coded Image

Figure 6. GIRL Image coded at three different rates

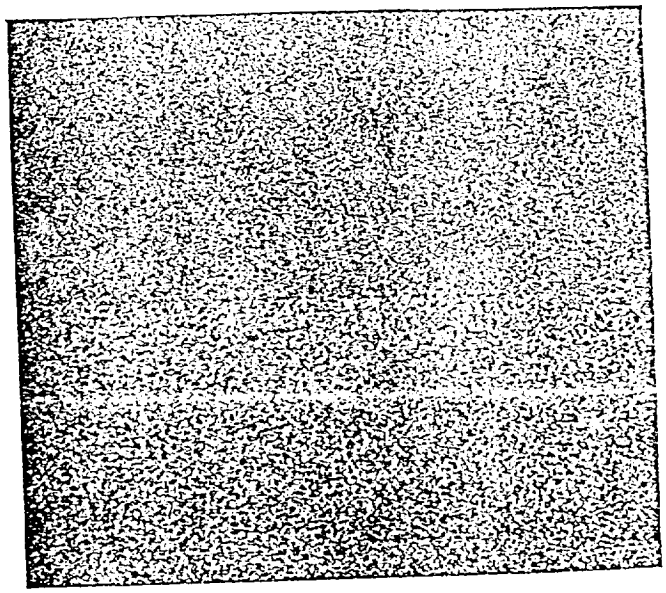


Figure 7. Error Image for GIRL image coded at three different rates

Appendix 2- Item 8

A ROBUST COMPRESSION SYSTEM FOR LOW BIT RATE TELEMETRY - TEST RESULTS WITH LUNAR DATA

Khalid Sayood and Martin C. Rost
Department of Electrical Engineering
University of Nebraska

PROBLEM STATEMENT

The output of a Gamma Ray detector is quantized using a 14 bit A/D converter. The number of each of the 2^{14} or 16,384 levels occurring in a 30 second interval is counted. In effect, a histogram of the gamma ray events is obtained with 16,384 bins. The contents of these bins are to be encoded without distortion and transmitted at a rate less than or equal to 600 bits per second. Thus the contents of the 16,384 bins are to encoded using 18000 bits. The encoder should be simple to implement and require only a minimal amount of buffering.

PROPOSED SYSTEM

Encoder

The contents of the bins are treated as a sequence for purposes of encoding. The proposed system encoder can be divided into two stages (three if a Huffman coding option is used. See Figure 1.) The first stage is a leaky differencer whose input/output relationship is given by

$$z_n = x_n - [\, a x_{n-1} \,]$$

where [t] is the largest integer less than or equal to t. The reason for using a leaky differencer is to allow the effect of errors to die out with time.

The output of the differencer forms the input for the second stage which is a modified runlength encoder. The encoder codebook contains six different types of symbols.

Mn - symbol used to represent negative differencer output values, for example, the differencer output values -1, -2,...,-n, are represented by the symbols M1, M2,...,Mn, respectively.

Pn - symbols used to represent positive differencer values, they are coded similar to the Mn symbols. Thus a differencer output value of +3 would be represented by the symbol P3.

Zn - symbols used to represent string of zeros of length n. Since the number of Z-symbols is kept small, these symbols represent "short" string of zeros (0-strings), while the S0- and S1-symbols to be introduced later represent "long" 0-strings.

BR - In the encoding scheme that follows, there will sometimes be a need to specify the end of a sequence. The BR or break symbol is used for this purpose.

S0XX - symbol used to represent long 0-strings. The S0 symbol indicates that a 0-string is being represented while X stands for a four bit word. XX is thus an eight bit word specifying the length of the 0-string.

S1XX - symbol used to represent long 0-strings that are followed by a 1. It is constructed in the same manner as the S0XX symbol.

Each symbol, Mn, Pn, Zn, BR, S0, and S1 is represented by a four bit word. The number of symbols in the encoder codebook is o(M)+o(P)+o(Z)+3 where o(M), o(P), and o(Z) are, respectively, the number of negative source symbols, positive source symbols, and short 0-strings symbols to be channel coded. As each symbol is represented by 4 bits, a total of sixteen encoder symbols are possible. In our coding scheme, o(M) is set to 2, o(Z) to 6, and o(P) to 5.

This means that if the differential output is -1, -2, 1, 2, 3, 4, 5 or a string of zeros of length five or less, it can be represented by a single symbol. What if the differential output is a positive value larger than five or a negative value less than -2? In such cases the largest (in magnitude) Mn or Pn symbol is used as a concatenation symbol. As an example, consider encoding the value 18.

Since o(P) is 5, the largest positive value that can be coded with a single symbol is 5. If P5 is also used as a concatenation symbol, larger source values can be coded. In this case, 18 can be coded as P5 P5 P5 P3. The receiver accumulates a total for all the P5 symbols consecutively received until a non-P5 symbol is received. This symbol is used to complete the current source value. In this case, P3 indicates the source value is 18.

In the case where the source value is a multiple of the maximum P-symbol value some confusion can occur in the decoding process. Consider the coding of the source values 10 followed by 8. In this case, four source symbols are required to code these values but, the receiver decodes them as a 18. To overcome this problem the break symbol (BR) is used. This symbol carries no data value but, is used by the receiver to prematurely stop the accumulation of P-symbols. Specifically, 10 and 8 are coded as P5 P5 BR P5 P3. The receiver stops constructing the first source value when the BR is encountered and start constructing the next with the following P5 symbol.

If a source value to coded is negative, the above procedure is used with the allowed M-symbols along with the BR symbol to prevent incorrect receiver decoding. For example, -3 would be encoded as M2M1 and -4 would be encoded as M2M2BR.
In this particular application, the tails of a given signal frame contain long runs of zeros that are separated by non-zero data values. It is very likely that these 0-string separators take the value 1. Thus, it is beneficial to code these runs with one of the following two symbols, each of which is three code words in length:

S0 x y a 0-string of length xy (base 16).

S1 x y a 0-string of length xy (base 16) followed a 1.

For example, the symbol, S0 4 0, represents a string of 64 0s, and the symbol, S1 4 0, represents a string of 64 0s followed by a 1. If the separating data value is not 1, then additional source symbols follow the S0 symbol to complete the description of its value. The maximum length of 0-string that can be coded with this type symbol is 255 (FF base 16). If a string of length greater than 255 is encountered, a concatenation rule must be applied.

Since the symbols S0 0 0 and S1 0 0 are not assigned, they are used as 0-string concatenation symbols. They are used to indicate the fact that a 0-string is to be built whose length is greater than 255. Each time one of these symbols is used it is assumed that a 0-string of length greater than 255 is being coded, and additional information is to be provided on its length by the following symbols. A 0-string is terminated if the last S0-symbol indicates a length value other than 00 for xy.

For example, if a 0-string of length 300 is followed by a 1, two source symbols (six channel words) are required to code the string: S1 0 0 S1 2 D. The value for xy of the first symbol is 00, so the 0-string is continued using the following S1-symbol(s). In this way, 0-

strings of arbitrary length can be constructed by concatenating as many S1 0 0 symbols as needed to bring the overall reconstructed 0-string length to within 255 0s of its full length. The final S1-symbol in such a series which does not have a 0 0 length indicator terminates the 0-string concatenation process. Since the S1 symbol is being used this 0-string is automatically followed by a 1. Consider coding a 0-string of length 300 that is followed by a -1. Two S0-symbols (six channel words) are required to code the 0-string, and one M-symbol (one channel word) is required to code the -1: S0 0 0 S0 2 D M1 for a total of seven channel words.

Since the long runlength symbols require three channel words each, an excessive amount of channel capacity can be wasted when coding short runs of 0s. As a consequence, a group of short run symbols that use only one channel word each are used to alleviate this problem. The identifier for these symbols is Zn (where n represents the length of the 0-string). For example, a run of 5 0s is represented by the symbol Z5. The coding length of a short 0-string using Zn symbols only improves the overall coding rate if the short 0-string is coded with fewer channel bits when using the Z-symbols instead of the S0- and S1-symbols.

Consider the following example for coding a string of 10 0s. Since o(Z) is 6, to code this 0-string using Z-symbols takes two channel words: Z6 Z4. But, when coded using an S0-symbol it takes three channel words to code this 0-string: S0 0 A. Therefore, the Z-symbol coding is more channel efficient. Since an S0- (or S1-) symbol always require three channel words, the only way to guarantee that short 0-strings are coded efficiently is to set the maximum number of short Z-symbols in a single 0-string coding to two. Thus, for an o(Z) of 6, the maximum 0-string length to be Z-symbol coded is 12.

The encoder described above has two main characteristics. First, it has been designed for the specific task noted in the problem statement. No claims are made regarding its suitability for other

tasks. The second characteristic is its simplicity. The encoding operation requires a very small amount of computation. Furthermore, the onboard memory requirements for buffering are minimal.

If Huffman coding is to be used, the final stage of the encoder is a Huffman coder. This will, of course, increase the complexity of the encoder and may make the system more vulnerable to channel errors. Therefore, if at all possible we will avoid using a Huffman coder.

Decoder

The decoder for the proposed system consists of three stages. The first stage of proposed system decoder is maximum A Priori Probability (MAP) receiver[6]. The MAP receiver design is based on the assumption that the output of the encoder contains dependencies.

The MAP design criterion can be formally stated as follows: For a discrete memoryless channel (DMC), let the channel input alphabet be denoted by $A = \{a_0, a_1, \ldots, a_{M-1}\}$, and the channel input and output sequences by $Y = \{Y_0, Y_1, \ldots, Y_{L-1}\}$ and $\hat{Y} = \{\hat{Y}_0, \hat{Y}_1, \ldots, \hat{Y}_{L-1}\}$, respectively. If $A = \{A_i\}$ is the set of sequences $A_i = \{\alpha_{i,0}, \alpha_{i,1}, \ldots, \alpha_{i,L-1}\}, \alpha_{i,k} \epsilon A$, then the optimum receiver (in the sense of maximizing the probability of making a correct decision) maximizes P[C], where

$$P[C] = \sum_{A_i} P[C|\hat{Y}]P[\hat{Y}].$$

This in turn implies that the optimum receiver maximizes $P[C|\hat{Y}]$. When the receiver selects the output to be A_k, then $P[C|\hat{Y}] = P[Y = A_k|\hat{Y}]$. Thus, the optimum receiver selects the sequence A_k such that

$$P[Y = A_k|\hat{Y}] \geq P[Y = A_i|\hat{Y}] \quad \forall_i.$$

When the channel input sequence is independent, this simplifies to the standard MAP receiver[6]. Under conditions where this is not true, the receiver becomes a sequence estimator which maximizes the path

metric. $\Sigma \log P(y_i|\hat{y}_i, y_{i-1})$ [5]. The path metric can be computed for a particular system by rewriting it using the following relationship[4].

$$P[y_i = a_j|\hat{y}_i = a_n, y_{i-1} = a_m] = \frac{P[\hat{y}_i = a_n|y_i = a_j]P[y_i = a_j|y_{i-1} = a_m]}{\Sigma_l\, P[y_i = a_l|y_{i-1} = a_m]P[\hat{y}_i = a_n|y_i = a_l]}$$

Notice that the right hand side consists of two sets of conditional probabilities $\{P[\hat{y}_i|y_i]\}$ and $\{P[y_i|y_{i-1}]\}$. The first set of conditional probabilities are the channel transition probabilities while the second depend only on the encoder output. The two are combined according to the above relationship to construct an $M \times M \times M$ lookup table for use in decoding. The structure of the MAP receiver is that of the Viterbi decoder[4,5].

The second stage of the decoder is the inverse operation of the modified run-length encoder. The operation of this stage has already been described in the previous section. The final stage of the decoder is the inverse of the differential operation with an input output relationship

$$x_n = z_n + [ax_{n-1}]$$

RESULTS

In this section we present results obtained by using the proposed system of the previous section. The data used was provided by Ms.\ M. Mingarelli-Armbruster of the Goddard Space Flight Center. This data was generated according to a Poisson distribution where the Poisson parameter was obtained from ten hours of lunar data. Both noisy and noiseless channel performance of the proposed system were examined via Monte-Carlo simulation. A total of twenty, 30-second intervals were used in the tests. The performance was compared with the Rice algorithm[1-3].

Before proceeding with the results, some caveats are in order. First,

the name Rice algorithm is a misnomer. What is presented[1-3] is not an algorithm but an approach. In this approach, a suite of algorithms is used to encode sections of the data, and the most efficient algorithm for that particular section of data is selected. In this way, data with very different statistical profiles can be accommodated. Thus what is presented[1-3] could more correctly be called the Rice Universal Coding Approach (RUCA). What we compare against here are algorithms presented[1-3] as examples of the RUCA. These algorithms were constructed for use in very general situations. As opposed to this, the particular algorithm presented here has been designed for a specific task. A final observation is that the encoder presented in this paper could very easily be used as the first stage of the RUCA. However, this would result in a rather complex encoder and substantial increase in the need for onboard memory over the proposed design. Therefore, if the algorithm presented in the previous section satisfies the requirements in terms of rate and robustness, such a step would be undesirable.

The results of the tests with both algorithms are presented in Table 1 and Table 2. The number of bits required to code twenty thirty-second intervals and the average rate needed for both algorithms is presented in Table 1. The second and third columns contain the total number of bits and the rate when the Rice algorithm is used. The average rate over twenty intervals is 719 bits per second. Columns three to six present the results obtained by using the proposed algorithm. The first two columns contain the results for the case where the Huffman coder was not used while the last two columns contain the results for when the Huffman coder formed the last stage of the encoder. The rate without the Huffman coder averaged over twenty intervals is 595 bits per second while the average rate when the Huffman coder is used is 522 bits per second. These results indicate that the proposed system will satisfy the specifications (coding rate below 600 bits per second) both when the Huffman coder is used and when it is not. As both systems meet the target and as the inclusion of the Huffman coder increases both the complexity and the vulnerability of the system to

channel noise, we elected to use the system without the Huffman coder.

Table 2 provides the performance of the algorithms under noisy channel conditions. Three performance measures are used, namely, mean squared error (MSE), mean absolute error (MAE), and the number of decoded values which are in error. Note the very large difference between the performance of the Rice algorithm and the proposed algorithm. Also, the proposed algorithm maintains a robust performance at extremely high error rates. In fact, under even highly adverse conditions the mean squared error is almost constant, and the number of erroneous decoded values is about 25% of the total. However, the performance of the algorithms at high error rates may be irrelevant in this particular situation. The reason being that the transmitted data will be well protected by a channel coding scheme consisting of a Reed-Solomon coder followed by a convolutional coder. This combination is expected to keep the average probability of error on the coded channel below 9×10^{-6}.

Finally, we examine the relative complexity and buffer requirements for the two algorithms. The proposed algorithm can be easily realized with a simple program implemented using a microprocessor. Based on the memory requirements for the simulation program used in this study, the memory needed for actual implementation should be about 1 K. The only time buffering may be required is when a large differencer output is encountered, and the encoder has to generate several channel symbols for one input. Depending on the way the entire system is implemented, the buffer requirements could range from a single symbol buffer to perhaps a sixteen symbol buffer.

As opposed to this, the Rice algorithm by its very nature, being a universal coding algorithm, is quite complex. Each block of data is encoded using a number of candidate algorithms; the algorithm which provides the most efficient encoding is then selected. Each of the candidate algorithms is itself relatively complex though some very ingenious techniques are used to make subunits of one algorithm common

to several candidate algorithms. Because several passes are required to do the encoding, the buffering requirements for this approach are substantial.

These differences in complexity are very natural based on the different objectives of the two algorithms. The proposed system is designed for a very specific situation while the Rice algorithm is designed to handle general situations.

TABLE 1

Coding rates with the Rice Algorithm and the Proposed Algorithm, (HC) denotes the results for the case where the Huffman Coder was used.

	RICE ALGORITHM		PROPOSED ALGORITHM			
INTERVAL	TOTAL BITS	RATE	TOTAL BITS	RATE	TOTAL BITS (HC)	RATE (HC)
1	21,647	721.6	17,832	594.4	15,733	524.4
2	21,385	712.8	17.528	584.3	15,345	511.5
3	21,530	717.7	17,784	592.8	15,520	517.3
4	21,562	718.7	17,840	594.7	15,691	523.0
5	21,666	722.2	18,144	604.8	15,883	529.4
6	21,424	714.1	17,504	583.5	15,457	515.2
7	21,841	728.0	18,048	601.6	15,882	529.4
8	21,630	721.0	18,096	603.2	15,907	530.2
9	21,719	723.9	18,132	604.4	15,843	528.1
10	21,568	718.9	18,096	603.2	15,695	523.2
11	21,308	710.3	17,604	586.8	15,438	514.6
12	21,509	716.9	17,728	590.9	15,580	519.3
13	21.633	721.1	17,780	592.7	15,581	519.4
14	21,822	727.4	18,016	600.5	15,913	530.4
15	21,296	709.8	17,564	585.4	15,361	512.0
16	21,701	723.4	17,956	598.5	15,872	529.1
17	21,058	701.9	17,296	576.5	15,139	504.6
18	21,312	710.4	17,688	589.6	15,449	514.9
19	21,713	723.8	18,160	605.3	16,033	534.4
20	21,888	729.6	18,292	609.7	16,125	537.5
OVERALL AVERAGE		718.7		595.1		522.4

TABLE 2

Performance of the algorithms under noisy channel conditions.

	RICE ALGORITHM		
PROBABILITY OF ERROR	MEAN SQUARED ERROR	MEAN ABSOLUTE ERROR	# OF DECODED ERRORS
10^{-6}	0.0760	0.023	140
10^{-5}	4.07	0.45	1,908
10^{-4}	31.49	3.14	10,177
10^{-3}	479.22	16.03	15,658
10^{-2}	8,562.87	76.75	16,189

	PROPOSED ALGORITHM		
PROBABILITY OF ERROR	MEAN SQUARED ERROR	MEAN ABSOLUTE ERROR	# OF DECODED ERROR
10^{-6}	2.4×10^{-5}	1.2×10^{-5}	1
10^{-5}	0.026	0.016	218
10^{-4}	0.17	0.14	1,287
10^{-3}	0.78	0.28	2,944
10^{-2}	6.81	0.71	3,765

SUMMARY AND CONCLUSIONS

We have presented a robust noiseless encoding scheme for encoding the gamma ray spectroscopy data. The encoding algorithm is simple to implement and has minimal buffering requirements. The decoder contains error correcting capability in the form of a MAP receiver. While the MAP receiver adds some complexity, this is limited to the decoder. Nothing additional is needed at the encoder side for its functioning.

ACKNOWLEDGMENT

This work was supported by a grant from the Goddard Space Flight Center, Greenbelt, Maryland, under Grant NAG5-916.

REFERENCES

1) R. F. Rice, "Practical Universal Noiseless Coding," 1979 SPIE Symposium Proceedings, Vol.\ 207, San Diego, CA, August 1979, pp. 247-267.

2) R. F. Rice, "Some Practical Universal Noiseless Coding Techniques," JPL Publication 79-22, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, March 15, 1979.

3) R. F. Rice and Jun-Ji Lee, "Some Practical Universal Noiseless Coding Techniques, Part II," JPL Publication 83-17, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, March 1, 1983.

4) K. Sayood and J. C. Borkenhagen, "Use of Residual Redundancy in the Design of Joint Source/Channel Coders," Submitted to IEEE Trans.\ Commun.

5) K. Sayood and J. D. Gibson, "Maximum Aposteriori Joint Source/Channel Coding," Proceedings of the 22nd Annual Conference on Information Sciences and Systems, Princeton, New Jersey, March 1988.

6) J. M. Wozencraft and I. M. Jacobs, "Principles of Communication Engineering. John Wiley and Sons, Inc., New York, 1965.

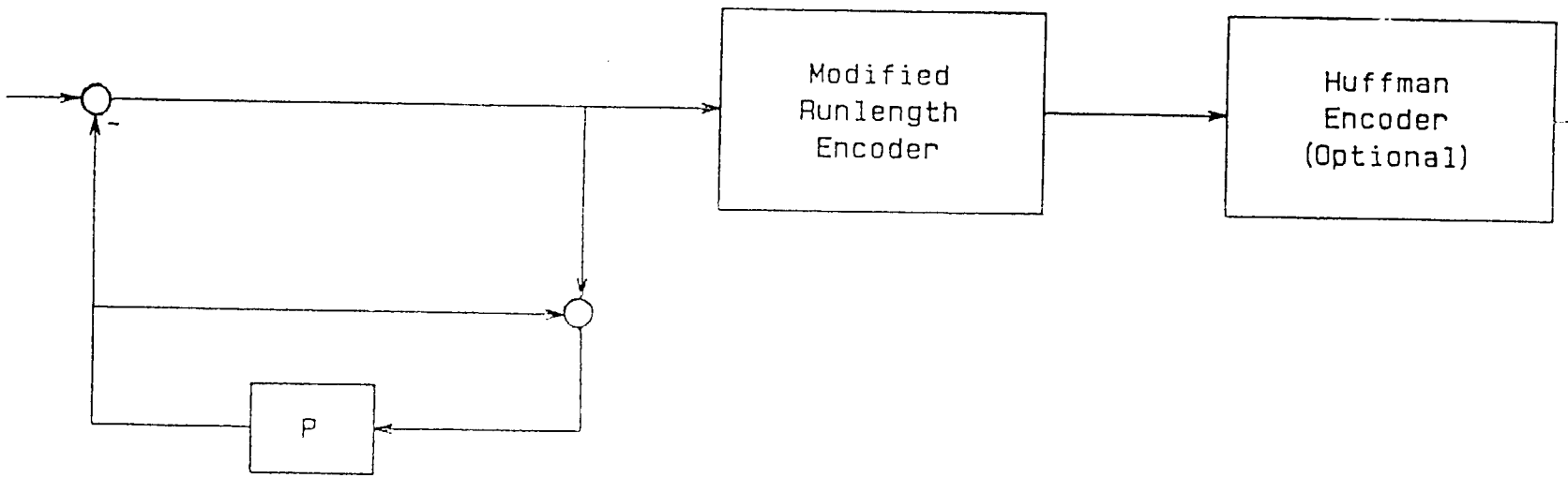


Figure 1. Proposed Encoder

Appendix 2- Item 9

edited by E. Arikan
Elsevier Science Publishers B.V., 1990

A ROBUST LOW-RATE CODING SCHEME FOR PACKET VIDEO

Y.C. Chen, K. Sayood and D. J. Nelson
Department of Electrical Engineering
Center for Communication and Information Science
University of Nebraska-Lincoln

I. INTRODUCTION

Due to the rapidly evolving field of image processing and networking, video information promises to be an important part of tomorrow's telecommunication system. Up to now, video transmission has been mainly transported over circuit-switched networks. It is quite likely that packet-switched networks will dominate the communications world in the near future. Asynchronous transfer mode (ATM) techniques in broadband-ISDN can provide a flexible, independent and high performance environment for video communication. Therefore, it is necessary to develop techniques for video transmission over such networks.

The recent literature contains a number of proposed packet video schemes. Verbiest and Pinnoo proposed a DPCM-based system which is comprised of an intrafield/interframe predictor, a nonlinear quantizer, and a variable length coder[1]. Ghanbari has simulated a two-layer conditional replenishment codec with a first layer based on hybrid DCT-DPCM and second layer using DPCM[2]. Darragh and Baker presented a sub-band codec which attains a user-prescribed fidelity by allowing the encoder's compression rate to vary[3]. Kishino et al. describe a layered coding technique using discrete cosine transform coding, which is suitable for packet loss compensation[4]. Karlsson and Vetterli presented a sub-band coder using DPCM with a nonuniform quantizer followed by run-length coding for baseband and PCM with run-length coding for nonbaseband[5]. In this paper, a different coding scheme called MBCPT is investigated. Unlike the methods mentioned above, MBCPT doesn't use decimation and interpolation filters to separate the signals into sub-bands. However it does have the attractive property of dealing separately with high frequency and low frequency information. This separation is obtained by the use of variable blocksize transform coding[6].

To deliver packets in a limited time and provide a real time service is a difficult resource allocation and control problem, especially when the source generates a high and greatly varying rate. In packet-switching networks, packet losses are inevitable, but use of a packet-switching network yields a better utilization of channel capacity. The video coder will require different channel capacity over time but the network will provide a channel whose capacity changes depending on the traffic in the network.

Therefore, the interactions between the coder and the network have to be considered and be incorporated into the requirements for the coder. These requirements include:

1. Adaptability: The video source has a varying information rate. The encoder should therefore generate different bit rates corresponding to the varying information rate.
2. Insensitivity to error: The coding scheme has to be robust to packet loss so as to preserve the image quality. Recall that retransmission is impossible because of tight timing requirements.
3. Control of coding rate: Sensing the heavy traffic in the network, the coding scheme is required to adjust the coding rate. In the case of a congested network, the coder could be switched to another mode which generates fewer bits with a minimal degradation of image quality.
4. Parallel architecture: The coder should preferably be implemented in parallel. This would allow the coding procedure to be run at a lower rate in many parallel streams.

In the next section, we investigate the proposed coding scheme to see how well it satisfies the above requirements.

II. MIXTURE BLOCK CODING WITH PROGRESSIVE TRANSMISSION

Mixture Block Coding (MBC) is a variable-blocksize transform coding algorithm which codes the image with different blocksizes depending upon the complexity of that block area.[6] When using MBC, the image is first divided into maximum blocksize blocks. After coding, the distortion between the reconstructed and original block is calculated. The block being processed is subdivided into smaller blocks if that distortion fails to meet the predetermined threshold. The coding-testing procedure continues until the distortion is small enough or the smallest blocksize is reached.

MBCPT is a multipass scheme in which each pass deals with different blocksizes. The first pass codes the image with maximum blocksize and transmits it immediately. Only those blocks which fail to meet the distortion threshold go down to the second pass which processes the difference image block, coming from the original and coded image obtained in the first pass, with smaller blocks. The difference image coding scheme continues until the final pass which deals with the minimum size block. At the receiving end, a crude image is obtained from the first pass in a short time and the data from following passes serve to enhance it. The coding structure is similar to a quad tree structure proposed by Dreizen[7], and Vaisey and Gersho[8]. In the quad tree coding structure used in this paper, a 16x16 block is coded and the distortion of the block is calculated. If the distortion is greater than the predetermined threshold for 16x16 blocks, the block is divided into four 8x8 blocks for additional coding. This coding-checking procedure is continued until the only image blocks not meeting the threshold are those of size 2x2.

The coding technique used is the discrete cosine transform. For all blocksizes, only four coefficients of the transform, including the dc and three lowest order frequency coefficients, are coded and others are set to zero. The dc coefficient in the first pass is coded with an 8-bit uniform quantizer. In the remaining passes it has a laplacian distribution and a 5-bit optimal laplacian nonuniform quantizer is used. As an alternative, an LBG vector quantizer with a 512 codebook size is used to quantize the vector which

comprises the three ac coefficients. The threshold of each pass has to be pre-selected and is readjustable during the operation according to the channel condition and quality required. Because only partial blocks which fail to meet the distortion threshold need to be coded, there must be some side information to instruct the receiver how to reconstruct the original image. One bit of overhead is needed for each block. If a block is to be divided, a 1 is assigned to be its overhead; if not, a 0 is assigned.

The interframe coder used in this paper is a differential scheme. This coder processes the difference image coming from the current frame and the previous frame which is locally decoded using data from the first three passes. Only the data from the first three passes is used because, while under conditions of no packet loss there is almost no difference between using three passes and using all four passes, there is substantially less degradation in the former approach when there is packet loss. This can be seen from the results in Fig. 1. In this paper, the Kronkite motion sequence with 16 frames is used as the simulation source. Every image consists of 256x256 pixels with graylevels ranging from 0 to 255. It is similar to a video conferencing type image which has neither rapid motion nor scenes changes. No motion detection or motion compensation techniques are used but could be implemented when broadcasting video.

From the datastream output listed in Table 1, we can see that the data in pass 4 represents 30-40% of the entire data. Pass 4 is primarily responsible for the clarity of the image and is usually labeled with the lowest priority in the network. We therefore call this the least significant pass(LSP). The packets containing this data have substantial possibility of being discarded due to low priority. As they are not used in the prediction process, their loss does not cause error propagation.

III. INTERACTION OF THE CODER AND THE NETWORK

The network simulator used for this study was a modified version of an existing simulator developed by Nelson et al[9]. Details of this simulator can be found in [10, 11]. When the video data is packed and sent into a nonideal network, some problems emerge.

A. Packetization

The task of the packetizer is to assemble video information, coding mode information, if it exists, and synchronization information into transmission cells. In order to prevent the propagation of the error resulting from the packet loss, no data from the same block or same frame is separated into different packets. As the segmentation process in the transport layer has no information regarding the video format, the packetization process has to be integrated with the encoder, which is in the presentation layer on the user's premise. Otherwise, some overhead has to be added into the datastream to guide the transport layer to perform the packetization in the desired manner.

Every packet must contain an absolute address which indicates the location of the first block it carries. Because every block in MBCPT has the same number of bits in each pass, there is no need to indicate the relative address of the following blocks contained in the same packet. Fixed length packetization is used in this paper for simplicity.

B. Error Recovery

There is no way to guarantee that packets won't get lost after being sent into the network. Packet loss can be mainly attributed to bit errors in the address field, leading the packets astray in the network, or due to congestion which exceeds the networks management ability causing packets to be discarded. Effects created by higher pass packet loss (like pass 4) in MBCPT coding will be masked by the basic passes and replaced with zeros. The distortion is almost invisible when viewing at video rates because the lost area is scattered spatially and over time. However, low pass packets loss (like pass 1), though rare due to high priority, will create an erasure effect which can be quite objectionable. Replacing lost data with reconstructed values from the corresponding area in the previous frame is not very effective. Motion detection and motion compensation could be used to find a best matched area for replacement in the previous frame.

Side information in the MBCPT decoding scheme is very important. To prevent this vital information from getting lost, error control coding can be applied in both directions along with and perpendicular to the packetization. The former is for bit error in the data field while the latter is for packet loss. The minimum distance that the error control coding should provide depends on the network's probability of packet loss, correlation of such loss and channel bit error rate. Also, as the output rate of side information and pass 1 and even pass 2 is quite steady, a fixed amount of channel capacity could be allocated to these outputs to ensure their timely arrival. That means circuit-switching can be used for important and steady data.

C. Flow Control

In order to shield the viewer from severe network congestion. Flow control schemes can be used. If the encoder is aware of congestion in the network, it can adjust its coding scheme to reduce the output rate. In the MBCPT coding scheme, if the output buffer exceeds a given threshold, the encoder can switch to a coarse quantizer with fewer steps or loosen the threshold to decrease its output rate. In this way, smooth quality degradation is obtainable. Of course, this also complicates the encoder design. It is also possible to use the congestion control of the network protocols to prevent the drastic quality change by assigning different priorities to packets from different passes. In the MBCPT coding scheme, side information and packets from pass 1 are assigned highest priority and higher pass packets are assigned decreasing priority.

D. Interaction with protocols

In the ISO model, physical, datalink and network layers comprise the lower layers which form a network node. The higher layers which consist of transport, session, presentation and application layers, typically reside in a customer's premises and perform all the functions of the packet video coder. The transport layer does the packetization and reassembly. The session layer supervises set-up and tear-down for sessions which have different types and quality. The quality of a set-up session can be determined by the threshold in the coding scheme and the priority assignment for transmission. Of course, the better the quality, the higher the cost. Fig. 2 shows the tradeoff between PSNR and video output rate by adjusting thresholds. The presentation layer does most of the signal processing, including separation and compression. Because it knows the video format exactly, if any error concealment is required, it will be performed here. The application layer works as a boundary between the user and the network and deals with all the analog-digital signal conversion.

IV. RESULTS FROM PACKET VIDEO SIMULATION

The results obtained in this packet video simulation show that a reasonable compression with good image quality can be obtained using the proposed scheme. The monochrome sequence used in this simulation corresponds to a bit rate of 15.3 Mbits/s, given a video rate of 30 frames/s. As Table 1 shows, the average data rates of our system is 1.539 Mbits/s. The compression rate is about 10 with a mean PSNR of 38.74 dB. Fig. 3 shows the data rate of the sequence frames with side information, 4 passes and total rate. It is clear that data rate of pass 1 is constant as long as the quantization mode is kept the same. Side information and data from pass 2, even pass 3, is quite steady. The data rate of pass 4 is bursty and highly-uncorrelated. Fig. 4 shows the PSNR for each frame in the sequence. The standard deviation is only 0.2 dB. In the simulation, the same threshold is used throughout the sequence. If constant visual quality is desired, a varying threshold can be used for different frames. That will generate a more variable bit rate and, of course, motion detection would be required.

From the difference images of this sequence, frames 1-8 seem quite motionless while frames 9-13 contain substantial motion. We adjusted the traffic condition of the network to force some of the packets to get lost so as to check the robustness of the coding scheme. Heavy traffic is set up in the motionless and motion period separately. The average packet loss percentage is 3.3% which is considered high for most networks. Fig. 5 shows an image which suffers packet loss from pass 4. As can be seen, the effect of lost packets is not at all severe, even if the lost packet rate is unrealistically high. Fig. 6 shows the case when packet loss occurs in pass 1. Clearly there are visible defects in the motion period. Apparently the replenishing scheme used here is not sufficient in areas with motion. It is believed that the performance can be improved with a motion compensator algorithm which would find the appropriate area for replenishment.

V. CONCLUSIONS

The network simulator was used only as a channel in this simulation. In fact, before the real-time processor is built, a lot of statistics can be collected from the network simulator to improve upon the coding scheme. These include transmission delays and losses from various passes under different network loads. For resynchronization, the delay jitter between received packets can also be estimated from this simulation. MBCPT has been investigated for use over packet networks and has been found to provide high compression rate with good visual performance, robustness to packet lost, tractable integration with network mechanics and simplicity in parallel implementation. For fast moving scenes, the differential MBCPT scheme seems insufficient. Motion compensation, error concealment or even attaching function commands into the coding scheme are believed to be useful tools to improve the performance and will be the direction of future research.

ACKNOWLEDGEMENT

This work was supported by a grant from the NASA Goddard Space Flight Center (NAG 5-916).

REFERENCES

[1] W. Verbiest and L. Pinnoo, "A Variable Bit Rate Codec for Asynchronous Transfer Mode Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 801–806, June 1989.

[2] M. Ghanbari, "Two-Layer Coding of Video Signals for VBR Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 801-806, June 1989.

[3] J. Darragh and R. Baker, "Fixed Distortion Subband Coding of Images for Packet-Switched Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 801–806, June 1989.

[4] F. Kishino, K. Manabe, Y. Hayashi and H. Yasuda, "Variable Bit-Rate Coding of Video Signals for ATM Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp.801–806, June 1989

[5] G. Karlsson and M. Vetterli, "Packet Video and Its Integration into Network Architecture," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 739–751, June 1989.

[6] M. C. Rost, "Data Compression Using Adaptive Transform Coding," Ph.D. Dissertation, University of Nebraska-Lincoln, 1988.

[7] H. M. Dreizen, "Content-Driven Progressive Transmission of Grey-Scale Images," *IEEE Trans. Commun.*, vol. COM-35, pp. 289-296, March 1987.

[8] D. Vaisey and A. Gersho, "Variable Block-Size Coding," *Proc. ICASSP*, pp. 1051–1054, April 1987.

[9] D. Nelson, K. Sayood, G. Ankenman and H. Chang, "Pnetsim System Programmer's Manual," University of Nebraska-Lincoln, Final Report for ARMY Contract DAAB07-85-K-K535, Dec. 1986.

[10] Y. C. Chen, "Mixture Block Coding with Progressive Transmission in Packet Video," MS Thesis, University of Nebraska-Lincoln, 1989.

[11] Y. C. Chen, K. Sayood, and D. J. Nelson, "A Packet Video Scheme Based On Variable Block-Size Coding," submitted to IEEE Transactions on Communications.

Table 1

	OVERHEAD	PASS1	PASS2	PASS3	PASS4	TOTAL
MEAN	65.28	130.56	214.50	591.87	538.86	1539.36
DEVIATION	8.70	00.00	32.82	95.37	210.00	311.85
MAXIMUM	77.04	130.56	280.56	735.84	821.52	1990.80
MINIMUM	44.88	130.56	136.08	384.72	21.84	1042.08

Output bit rate for each and total pass calculated with 30 frames/sec video rate. The maximum and minimum values are the instantaneous rates, which correspond to the respective maximum and minimum number of bits needed to encode a particular frame in the sequence. The unit is kilobits.

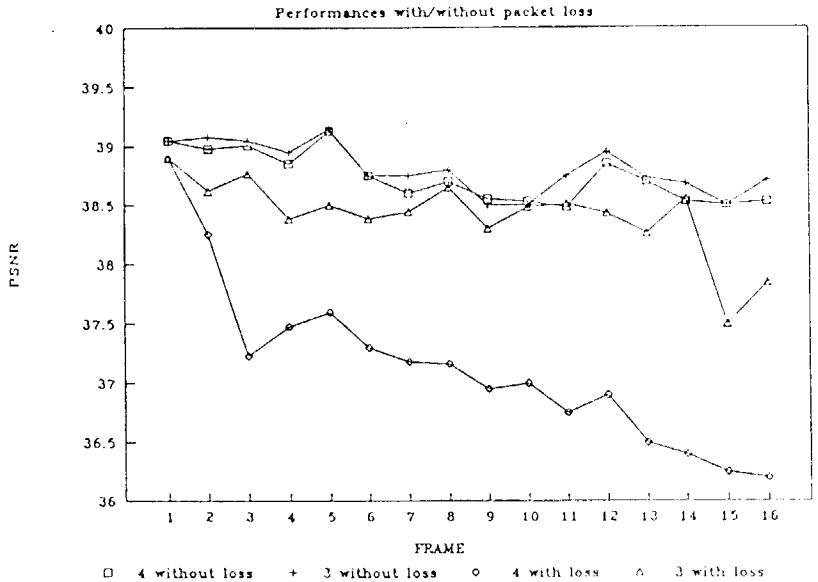


Figure 1 Performance with and without packet loss.

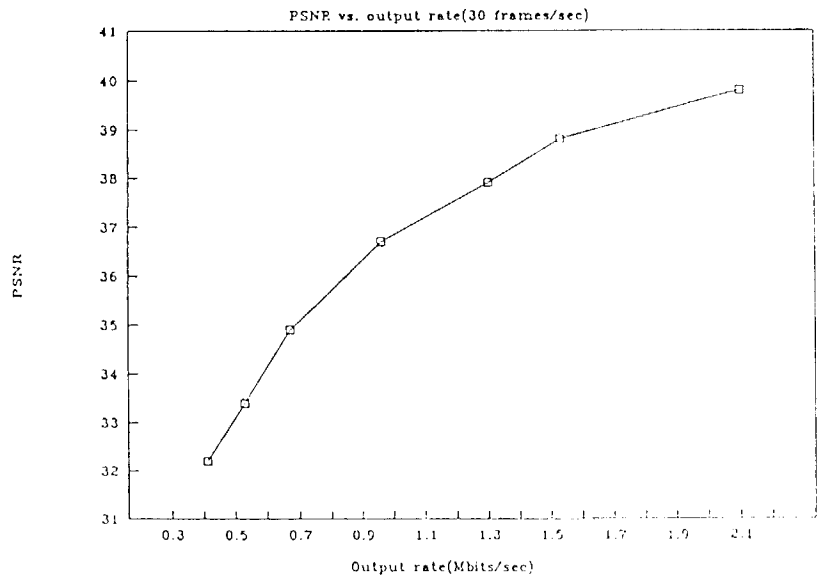


Figure 2 PSNR versus video output rate with 30 frames per second.

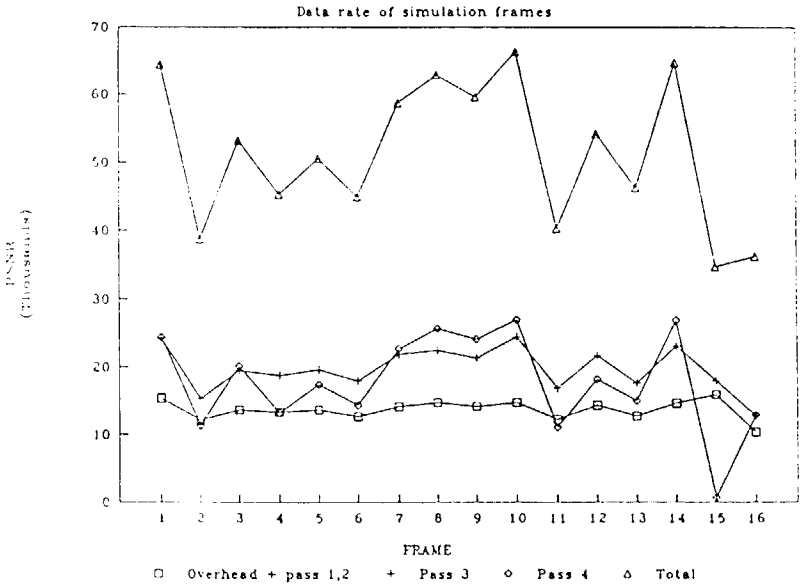


Figure 3 Data rate of simulation sequence frames.

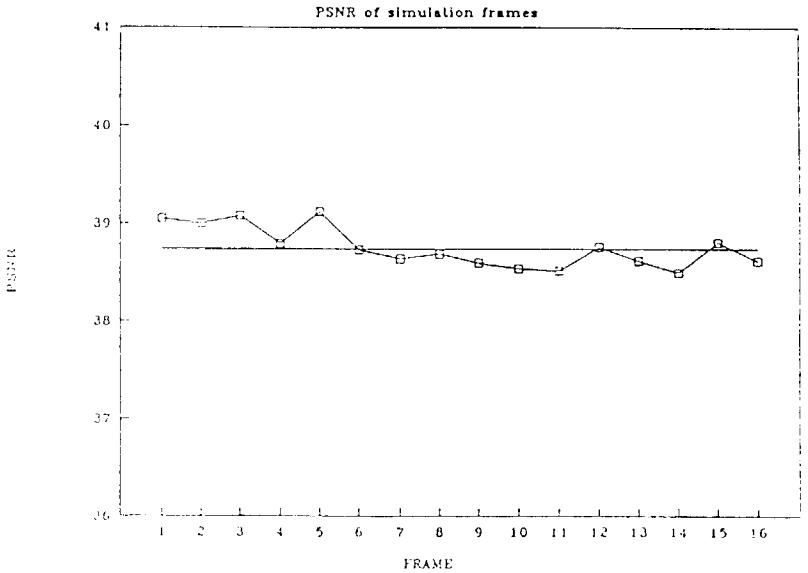


Figure 4 PSNR of simulation sequence frames.

Figure 5 The effect of pass 4 packet loss for frame 4.

Figure 6 The effect of pass 1 packet loss for frame 3.

Communication, Control, and Signal Processing
edited by E. Arikan
Elsevier Science Publishers B.V., 1990

A HYBRID LBG/LATTICE VECTOR QUANTIZER FOR HIGH QUALITY IMAGE CODING

V. Ramamoorthy[1] and K. Sayood[2]
[1]SBC Technology Resources Inc.
Southwestern Bell Corporation

[2]Department of Electrical Engineering
Center for Communication and Information Science
University of Nebraska-Lincoln

I. INTRODUCTION

It is well known that a vector quantizer (VQ) [1-4] is an efficient coder offering a good trade-off between quantization distortion and bit rate. The most popular vector quantizers are those constructed using the well known LBG algorithm [2]. The performance of a vector quantizer asymptotically approaches the optimum bound with increasing dimensionality. However in the case of LBG VQ, increasing the dimensionality causes the search and implementation complexity to be greatly increased and practical designs are therefore limited to low bit rates and small dimensionality. A vector quantized image suffers from the following types of degradations: a) Edge regions in the coded image contain staircase effects, b) Quasi-constant or slowly varying regions suffer from contouring effects, and, c) Textured regions lose details and suffer from granular noise. Staircase and contouring effects can immediately be spotted in an image; on the other hand, the effect of the granular noise is often mitigated by the very nature of the textured regions. All these three degradations are due to the finite size of the code book, the distortion measure used in the design and due to the finite training procedure involved in the construction of the code book. In this paper we present an adaptive technique which attempts to ameliorate the edge distortion and contouring effects.

In order to understand and evaluate the severity of the degradations caused by vector quantization, Ramamoorthy and Jayant performed several experiments by swapping regions in the coded image by the corresponding regions in the uncoded original[5]. When the coded edge regions were replaced with the corresponding regions of the uncoded original, the viewer failed to notice the granular and contouring distortions in the coded image. Careful examination over a long interval of time can indeed detect all the distortions present in the coded image. But a casual viewer who spent a few seconds of time in scrutinizing the coded image did not see the distortions. Given that edges are of such great importance, it makes sense that regions containing edges be quantized with higher fidelity. However, to do so generally requires a larger codebook, which is not feasible with an LBG VQ.

Appendix 2- Item 10

To appear in IEEE Transactions on Communications.

A Robust Coding Scheme for Packet Video

Y.C. Chen, K. Sayood and D. J. Nelson
Department of Electrical Engineering
Center for Communication and Information Science
University of Nebraska-Lincoln

Abstract

We present a layered packet video coding algorithm based on a progressive transmission scheme. The algorithm provides good compression and can handle significant packet loss with graceful degradation in the reconstruction sequence. Simulation results for various conditions are presented.

I. INTRODUCTION

Due to the rapid evolution in the fields of image processing and networking, video information will be an important part of tomorrow's telecommunication system. Up to now, video transmission has been mainly transported over circuit-switched networks. It is quite likely that packet-switched networks will dominate the communications world in the near future. Asynchronous transfer mode (ATM) techniques in broadband-ISDN can provide a flexible, independent and high performance environment for video communication. Therefore, it is necessary to develop techniques for video transmission over such networks.

The classic approach in circuit switching is to provide a "dedicated path," thus reserving a continuous bandwidth capacity in advance. Any unused bandwidth capacity on the allocated circuit is therefore wasted. Rapidly varying signals, like video signals, require too much bandwidth to be accommodated by a standard circuit-switching channel. With a certain amount of capacity assigned to a given source, if the output rate of that source is larger than the channel capacity, quality will be degraded. If the generating rate is less than the available capacity, the excess channel capacity is wasted. The use of packet networks allows for the utilization of channel sharing protocols between independent sources and can improve channel utilization. Another point that strongly favors packet-switched networks is the possibility that the integration of services in a network will be facilitated if all of the signals are separated into packets with the same format.

Some coding schemes which support packet video have been explored. Verbiest and Pinnoo proposed a DPCM-based system which is comprised of an intrafield/interframe predictor, a

This work was supported by a grant from the NASA Goddard Space Flight Center (NAG 5-916)

nonlinear quantizer, and a variable length coder[1]. Their codec obtains stable picture quality by switching between three different coding modes: intrafield DPCM, interframe DPCM, and no replenishment. Ghanbari has simulated a two-layer conditional replenishment codec with a first layer based on hybrid DCT-DPCM and second layer using DPCM[2]. This scheme generates two type of packets: "guaranteed packets" contain vital information and "enhancement packets" contain "add-on" information. Darragh and Baker presented a sub-band codec which attains a user-prescribed fidelity by allowing the encoder's compression rate to vary[3]. The codec's design is based on an algorithm that allocates distortion among the sub-bands to minimize channel entropy. Kishino et al. describe a layered coding technique using discrete cosine transform coding, which is suitable for packet loss compensation[4]. Karlsson and Vetterli presented a sub-band coder using DPCM with a nonuniform quantizer followed by run-length coding for baseband and PCM with run-length coding for nonbaseband[5]. In this paper, a different coding scheme based on a progressive transmission scheme called Mixture Block Coding with Progressive Transmission (MBCPT) [6,7] is investigated. Unlike those methods mentioned above, MBCPT doesn't use decimation and interpolation filters to separate the signals into sub-bands. However, it does have the attractive property of dealing separately with high frequency and low frequency information. This separation is obtained by the use of variable blocksize transform coding.

This paper is organized as follows. First, some of the important characteristics and requirements of packet video are discussed. In Section 3, the coding scheme called Mixture Block Coding with Progressive Transmission (MBCPT) is presented. In Section 4, a network simulator used in testing the scheme is introduced. In Section 5, the simulation results are discussed. Finally, in Section 6 the paper is summarized.

II. CHARACTERISTICS OF PACKET VIDEO

The demand for various services, such as telemetry, terminal and computer connections, voice communications, and full-motion high-resolution video, along with the wide range of bit rates and holding times they represent, provides an impetus for building a Broadband Integrated Service Digital Network (B-ISDN). B-ISDN is a projected worldwide public telecommunications network that will service a wide range of user needs. The continuing advances in the technology of optical fiber transmission and integrated circuit fabrication have been driving forces to realize B-ISDN. The idea of B-ISDN is to build a complete end-to-end switched digital telecommunication network with broadband channels. Still to be precisely defined by CCITT, with fiber transmission, H4 has an access rate of about 135 Mbps.

Packet-switched networks have the unique characteristics of dynamic bandwidth allocation for transmission and switching resources, and the elimination of channel structure. They acquire and release bandwidth as needed. Because the video signals vary greatly in bandwidth requirement, it is attractive to utilize a packet-switched network for video coded signals. Allowing the transmission rate to vary, video coding based on packet transmission permits the possibility

of keeping the picture quality constant, by implementing "bandwidth on demand". There are three main merits when transmitting video packets over a packet-switched network:

1. Improved and consistent image quality: if video signals are transmitted over fixed-rate circuits, there is a need to keep the coded bit rate constant, resulting in image degradation accompanying rapid motion.
2. Multimedia integration: as mentioned above, integrated broadband services can be provided using unified protocols.
3. Improved transmission efficiency: using variable bit-rate coding and channel sharing among multiple video sources, scenes can be transmitted without distortion if other sources, at the same time, are without rapid motion.

However video transmission over packet networks also has the following drawbacks:

1. The time taken to transmit a packet of data may change from time to time.
2. Packets may be delayed to the point where, because of constraints due to the Human Visual System, they have to be discarded.
3. Headers of packets may be changed because of errors and delivered to the wrong receiver.

It has to be emphasized that the delay/lost effect can reach very high levels if the combined users' requirement exceeds the acquirable bandwidth and may seriously damage the quality of the image.

When the signals transmitted in the network are nonstationary and circuit-switching is used with limited bandwidth, a buffer between the coder and the channel is needed to smooth out the varying rate. If the amount of data in the buffer exceeds a certain threshold, the encoder is instructed to switch into a coding mode that has lower rate but worse quality to avoid buffer overflow. In packet-switched networks, Asynchronous Time Division Multiplexing (ATDM) can efficiently absorb temporal variations of the bit-rate of individual sources by smoothing out the aggregate of several independent streams in the common network buffers[8].

To deliver packets in a limited time and provide a real time service is a difficult resource allocation and control problem, especially when the source generates a high and greatly varying rate. In packet-switched networks, packet losses are inevitable, but use of a packet-switched network yields a better utilization of channel capacity. However, it should be noted that the varying rate requirements of the video coder may not be synchronized with the variations in available channel capacity which changes depending on the traffic in the network. Therefore, the interactions between the coder and the network have to be considered and be incorporated into the requirements for the coder. These requirements include:

1. Adaptability of the coding scheme: The video source we are dealing with has a varying information rate. So it is expected that the encoder should generate different bit rates by removing the redundancy. When the video is still, there is no need to transmit anything.
2. Insensitivity to error: The coding scheme has to be robust to the packet loss so that the quality of the image is never seriously damaged. Remember that retransmission is impossible because of the tight timing requirement.

3. Resynchronization of the video: Because the varying packet-generating rate and the lack of a common clock between the coder and the decoder, we have to find a way to reconstruct the received data which is synchronous to the display terminal.
4. Control of coding rate: Sensing the heavy traffic in the network, the coding scheme is required to adjust the coding rate by itself. In the case of a congested network, the coder could be switched to another mode which generates fewer bits with a minimal degradation of image quality.
5. Parallel architecture: The coder should preferably be implemented in parallel. That allows the coding procedure to be run at a lower rate in many parallel streams.

In the next section, we investigate a coding scheme to see how well it satisfies the above requirements.

III. MIXTURE BLOCK CODING WITH PROGRESSIVE TRANSMISSION

Mixture Block Coding (MBC) is a variable-blocksize transform coding algorithm which codes the image with different blocksizes depending upon the complexity of that block area. Low-Complexity areas are coded with a large blocksize transform coder while high-complexity regions are coded with small blocksize. The complexity of the specific block is determined by the distortion between the coded and original image when the same number of bits are used to code each block. A more complex image block has higher distortion. The advantage of using MBC is that it does not process different complex regions with the same blocksize. That means MBC has the ability to choose a finer or coarser coding scheme to deal with different complex parts of the same image. With the same rate, MBC is able to provide an image of higher quality than a coding scheme which codes different complex regions with the same blocksize coder.

When using MBC, the image is divided into maximum blocksize blocks. After coding, the distortion between the reconstructed and original block is calculated. The block being processed is subdivided into smaller blocks if that distortion fails to meet the predetermined threshold. The coding-testing procedure continues until the distortion is small enough or the smallest blocksize is reached. In this scheme, every block is coded until the reconstructed image is satisfactory and then moves to the next block.

Mixture Block Coding with progressive transmission (MBCPT) is a coding scheme which combines MBC and progressive coding. Progressive coding is an approach that allows an initial image to be transmitted at a lower bit rate which can later be updated[9]. In this way, successive approximations converge to the target image with the first approximation carrying the "most" information and the following approximations enhancing it. The process is like focusing a lens, where the entire image is transformed from low-quality into high-quality. In progressive coding, every pixel value, or the information contained in it, is possibly coded more than once and the total bit rate may increase due to different coding scheme and quality desired. Because only the

gross features of an image are being coded and transmitted in the first pass, the processing time is greatly reduced for the first pass and a coarse version of the image can be displayed without significant delay. It has been shown that it is perceptually useful to get a crude image in a short time, rather than waiting a long time to get a clear complete image.

With different stopping criterion, progressive coding is suitable for dynamic channel capacity allocation. If a predetermined distortion threshold is met, processing is stopped and no more refining action is needed. The threshold value can be adjusted according to the traffic condition in the channel. Successive approximations (or iterations) are sent through the channel in progressive coding and lead the receiver to the desired image. If these successive approximations are marked with decreasing priority, then a sudden decrease in channel capacity may only cause the received image to suffer from quality degradation rather than total loss of parts of the images.

MBCPT is a multipass scheme in which each pass deals with different blocksizes. The first pass codes the image with maximum blocksize and transmits it immediately. Only those blocks which fail to meet the distortion threshold go down to the second pass which processes the difference image block, coming from the original and coded image obtained in the first pass, with smaller blocks. The difference image coding scheme continues until the final pass which deals with the minimum size block. At the receiving end, a crude image is obtained from the first pass in a short time and the data from following passes serve to enhance it. Fig. 1 shows the structure of pass consisting of 16x16 blocks for MBCPT. Fig. 2 shows the parallel structure of MBCPT. Coding algorithms using quad trees have also been proposed by Dreizen[10] and Vaisey and Gersho[11]. In the quad tree coding structure of this paper, the 16x16 block is coded and the distortion of the block is calculated. If the distortion is greater than the predetermined threshold for 16x16 blocks, the block is divided into four 8x8 blocks for additional coding. This coding-checking procedure is continued until the only image blocks not meeting the threshold are those of size 2x2. Figure 3 shows the algorithm.

The block size used in the coding scheme should be small enough for ease of processing and storage requirements, but large enough to limit the inter-block redundancy[12]. Larger block size results in higher compression, but it is very difficult to build real-time hardware for blocksizes larger than 16x16 because of the increase in the number of computations. So, 16x16 is chosen to be the largest blocksize. The minimum blocksize determines the finest visual quality that is achievable in the busy area. If the minimum blocksize is too large, it is possible to observe the blockiness in the coded edge of spherical objects because the coding block is square. In order to match the zonal transform coding used in this paper, 2x2 is the smallest blocksize and there are four passes (16x16, 8x8, 4x4, 2x2) in this scheme. Fig. 4-7 show images from the 4 passes.

After applying the discrete cosine transform, only four coefficients, including the dc and three lowest order frequency coefficients, are coded and others are set to zero. The dc coefficient in the first pass is coded with an 8-bit uniform quantizer due to the fact that it closely reflects the average gray level for that image block and is hard to model. The dc coefficient in the subsequent passes follows a laplacian model, and a 5-bit optimal laplacian nonuniform quantizer is used to also follow a laplacian model with a variance greater than that of the dc coefficient and can therefore also be coded using a laplacian quantizer. As an alternative, an LBG vector quantizer with a 512 codebook size is used to quantize the vector which comprises the three ac

coefficients. The initial threshold of each pass is selected beforehand and is readjustable during the operation according to the channel condition and quality required.

Because only partial blocks which fail to meet the distortion threshold need to be coded, side information is needed to instruct the receiver on how to reconstruct the image. One bit of overhead is needed for each block. If a block is to be divided, a 1 is assigned to be its overhead; if not, a 0 is assigned. The example shown in Fig. 8 has the following overhead: 1,1001,1001,1001,1001,1001.

The interframe coder used in this paper is a differential scheme which is based on MBCPT. This coder processes the difference image coming from the current frame and the previous frame which is locally decoded from the first three pass data. Fig. 9 shows the algorithm of this coder. Fig. 10 shows a different scheme which does the local decoding with all four passes. From Fig. 11, it can be seen that when there is no packet loss, the performances of these two schemes are quite the same. But when congestion occurs in the network, with the priorities assigned to packets, packets from pass 4 are expected to be discarded first. In this case, the performance (from Fig. 12) of the scheme in Fig. 9 is much better than the one in Fig. 10. Therefore the coding scheme in Fig. 9 is used in our simulation. In this paper, the Kronkite motion sequence from the USC database with 16 frames is used as the simulation source. Every image is 256x256 pixels with graylevels ranging from 0 to 255. It is similar to a video conferencing type image which has neither rapid motion nor scenes changes. Due to this characteristic, advanced techniques like motion detection or motion compensation have not been used but could be implemented when broadcasting video.

From the datastream output that is listed in Table 1, we can see that the data in pass 4 represents 30-40% of the entire data. This part of the data is involved in increasing the sharpness of the image and is usually labeled with the lowest priority in network. We therefore call this the least significant pass(LSP). With a substantial possibility of being discarded due to low priority, those packets from pass 4 won't be used to reconstruct the locally decoded image and be stored in the frame memory. This prevents the packet loss error propagating into following frames if the lost packet belongs to pass 4.

IV. SIMULATION NETWORK

The network simulator used for this study was a modified version of an existing simulator developed by Nelson et al.[13]. A brief description of the simulator is provided here.

A. Introduction

As mentioned in section 2, tomorrow's integrated telecommunication network is a very complicated and dynamic structure. Its efficiency requires sophisticated monitoring and control algorithms with communication between nodes reflecting the existing capacity and reliability of system components. The scheme for communicating information regarding the operating status

is called the system protocol. Since the communication of system information must flow through the channel, it reduces the overall capacity of the physical layers, but hopefully provides a more efficient system overall. Therefore, system efficiency depends entirely upon these protocols, which, in turn, depend upon the system topology, communication channel properties, nodal memory and component reliability. Most network protocols have been developed to provide high reliability in topological structures with reasonably high channel reliability.

In order to fit into the purpose of this study, most modifications which were made to the simulator were in those modules concerning the network layer. Since the simulator is structured in modules which represent, to some degree, the ISO Model for packet switched networks, a more detailed description about the network layer modules follows.

B. The Network Layer and Basic Operation

The simulation of a layer at each node is represented by a "processor" and one or more "packet queues." All events are scheduled through the "Sim_Q" which drives the simulator. Initially, the processors are all idle, the packet queues are all empty and the only tasks scheduled are the arrival of messages at the various nodes. The simulator operation occurs by examining the next event and performing the task indicated. The task may result in the scheduling of additional events, generally referred to as task completion times. When a message or packet is placed in the input queue at a node for a given layer, the processor for that queue is marked as busy, the packet is removed from the queue, and the task to be performed by the processor is scheduled for completion. When the task is completed (as a result of the simulator reaching that point in time), the "processor" examines the queue. If the queue is empty, the processor is set idle; otherwise it removes the next message or packet from the queue and schedules the completion of the operation which must be performed. The layers in the simulator are quite close in operation to the ISO transport, network and datalink layers.

(1) The Session Layer

In the OSI model, the session layer (SL) allows users to establish "sessions" on local or remote systems. In the simulator, as mentioned above, it contains a relatively simple model of the subscribers, participates in flow-control, and acts as a statistics collector for messages arriving and delivered. At message arrival time (from Sim_Q), the session layer generates the "message" with all of its randomly selected attributes and if flow control or node hold-down are not in effect, submits it to the transport layer. It then schedules the next message arrival time. During initialization, the task "SL_Rcv_Msg" for each node is queued in Sim_Q for the arrival time of the first message at that node. When this task is executed by the simulator, a message packet is generated and placed in the transport queue. The arrival of the next message is then queued in Sim_Q with the same task and with an arrival time determined by the random number generator (Poisson Distributed). The only other task performed by the session layer is the "SL_Snd_Msg" task that simulates delivery of messages to the subscribers, develops message statistics and "cleans up" the queues for messages delivered.

(2) The Transport Layer

The basic function of the transport layer at the sending end is to receive the message from the session layer, place it in packets and pass the packets on to the network layer. At the receiving end, the packets are reassembled into a message for delivery to the session layer. To accomplish the complex task of assuring reliable delivery, there is a transport time-out mechanism at both the sending and receiving nodes and a message acknowledgement packet that is sent to the sending node when all packets for the message have been satisfactorily received. At the sending end, if a message acknowledgment is not received in the allotted time period, the message can be retransmitted. In the simulations reported in this paper, the retransmission feature was not used. At the receiving end, if all packets are not received in the specified period of time, the entire message is discarded. It is recognized that in some networks, packetization takes place at the network level, leaving the transport layer responsible only for message-level structures. Reassembly, depending upon the protocol, can take place as low as the datalink level. These tasks were both placed in the transport layer, but are modular, and could be extracted and placed elsewhere. Also, the simulator was originally designed for datagram service, and since the packets do not necessarily arrive in order, it is unlikely that assembly would take place at the datalink level.

(3) The Network Layer

The network layer is concerned with controlling the operation of the network. A key design issue is determining how packets are routed from source to destination. Another issue is how to avoid the congestion caused when too many packets are presented to the network at the same time. In the simulator, the network layer performs all of the functions related to these two aspects with the exception of that aspect of flow control which takes place at the session layer, and the recovery protocols which require some service from the datalink layer. It also activates new channels when needed and determines when packets originating at other nodes are to be discarded. The network layer is currently the most dynamic with regard to the coding of modules. Five modules currently comprise the network layer. These include relatively static modules; one module for capturing lines or channels when more capacity is required and releasing them when they are not needed; one module for the network processor and queue handling and one module for the routines which are common to most routing algorithms. This leaves two modules for the dynamic parts of the routing and flow control algorithms.

(4) The Datalink Layer

The main task of the datalink layer is to take a raw transmission facility and transform it into a line or channel that appears free of transmission errors to the network layer. It simulates the sending of the message over the channel and the delivery at the other end. When a packet is received, the datalink acknowledgement is initiated either by the piggy-back acknowledgement or by generating a datalink acknowledgement packet. As mentioned previously, the datalink level also simulates the physical layer on a statistical basis. (Entered bit error rates are used in conjunction with a random number generator to determine if messages are corrupted.) When a line is "brought up", health packets are used to establish initial connections. Also, when a

line "goes down", an active node will immediately issue health check packets to ascertain when the channel is again available.

C. Modifications

A major problem of using this system as a simulation tool for the study of packet video is that as initially designed the system did not actually transmit messages from node to node. While a "packet" carrying all the necessary describing information moved from node to node, there no actual data in the packet. Therefore, modifications had to be made to the simulator to accommodate the video data. In the sending node, a field called "Image" which contains real image data is attached to the record "Packet_Ptr" allocated to the message generated in the session layer. There are three new modules in this layer. First, "Get_Image" puts the image data into the image field of a message generated at a specific time and node. Second, "Image_Available" checks to see if there is any image data that still needs to be transmitted. If that is true, the following message, generated at that specific node, is still the image message and contains some image data. Third, "Receive_Image" collects the image data in the session layer of the receiving node when the flag "Image_Complete" is on. In module "Session_Msg_Arrive", different priorities are assigned to different messages. In module "Session_Msg_ Send", some statistics are calculated including the number of lost image packets and the transmission delay for image packets.

In the original deisgn, the transport layer simply duplicated the same packet with different assigned sequential packet numbers without actually packetizing the message. The module "Transport_Packetize" has been modified to really packetize the image data which resides in the message record queued in "Transport_Q" when it is called. The module "Transport_Reassemble" is called to reassemble these image packets according to their packet number when the flag "Image_Content" defined in "Packet_Ptr" is true. The network layer is responsible for routing and flow-control. This module was already very well developed, so the modifications to be performed here were relatively minor. In the datalink layer, in order to simulate the delivery of packets through the channel, a new packet is generated at the receiving node and the information including the image data from the transmitted packet (which will still be resident at the sending node) are copied into it. Using existing bit-error-rates, the transmission success rate can be set and bit errors can be inserted in both the data and control bits in the packet. Errors in the control bits are simulated separately as long as the error rates are consistent. If an error in control bits occurs, the transmission is assumed to fail and retransmission will occur, again depending on the threshold of the timeout number. In addition to the modifications made to the layer modules, we had to arrange some new memory elements allocated for image messages and packets. In order to make sure the simulation is run in the steady state, the image data is made available to the network after some simulation time has passed.

V. INTERACTION OF THE CODER AND THE NETWORK

When the video data is packed and sent into a nonideal network, some problems emerge. These are discussed in the following section.

A. Packetization

The task of the packetizer is to assemble video information, coding mode information, if it exists, and synchronization information into transmission cells. In order to prevent the propagation of the error resulting from the packet loss, packets are made independent of each other and no data from the same block or same frame is separated into different packets. The segmentation process in the transport layer has no information regarding the video format. To avoid the bit stream being cut randomly, the packetization process has to be integrated with the encoder, which is in the presentation layer of the user's premise. Otherwise, some overhead has to be added into the datastream to guide the transport layer to perform the packetization in the desired manner. In order to limit the delay of packetization, it is necessary to stuff the last cell of a packet video with dummy bits if the cell is not completely full.

Every packet must contain an absolute address which indicates the location of the first block it carries. Because every block in MBCPT has the same number of bits in each pass, there is no need to indicate the relative address of the following blocks contained in the same packet. There always exists a tradeoff between packaging efficiency and error resilience. If error resilience is considerable, one packet should contain a smaller number of blocks. However, since each channel access by a station contains overhead, the packet length should be large for transmission efficiency. Fixed length packetization is used in this paper for simplicity.

Because of the structure of the coding scheme, the packets are classified into four priorities, with the packets from the first pass classified as the highest priority packets, and the packets from the fourth pass as the lowest priority packets.

This priority assignment also reflects the importance of the various packets to the reconstruction of the image sequence at the receiver. Table 1 shows the effect of approximately the same amount of packets lost in each pass on the reconstructed error in the received sequence.

B. Error Recovery

There is no way to guarantee that packets won't get lost after being sent into the network. Packet loss can be mainly attributed to two problems. First, bit errors can occur in the address field, leading the packets astray in the network. Second, congestion can exceed the networks management ability and packets are forced to be discarded due to buffer overflow. Effects created by higher pass packet loss (like pass 4) in MBCPT coding will be masked by the basic passes and replaced with zeros. The distortion is almost invisible when viewing at video rates because the lost area is scattered spatially and over time. However, low pass packets loss (like pass 1), though rare due to high priority, will create an erasure effect due to packetization and the effect is very objectionable.

Considering the tight time constraint, retransmission is not feasible in packet video. It may also result in more severe congestion. Thus, error recovery has to be performed by the decoder alone. In our differential MBCPT scheme, the packets from pass 4 are labeled lowest priority and form a great part of the complete data. These packets can be discarded whenever network congestion occurs. That will reduce the network congestion and won't cause too much degradation in quality. The erasures caused by basic pass loss are simply covered with the reconstructed values from the corresponding area in the previous frame. This remedy seems insufficient even when there is only small amount of motion in that area. Motion detection and motion compensation could be used to find a best matched area for replacement in the previous frame.

Side information in the MBCPT decoding scheme is very important. So, this vital information is not allowed to get lost. Two methods can be used for protection. First, error control coding, like block codes or convolutional codes, can be applied in both directions along with and perpendicular to the packetization. The former is for bit error in the data field while the latter is for packet loss. The minimum distance that the error control coding should provide depends on the network's probability of packet loss, correlation of such loss and channel bit error rate. Second, from Table 2, we can see that the output rate of side information and pass 1 and even pass 2 is quite steady. It seems feasible to reserve a certain amount of channel capacity to these outputs to ensure their timely arrival. That means circuit-switching can be used for important and steady data.

C. Flow Control

In order to shield the viewer from severe network congestion, there are some flow control schemes which are considered useful. If there is an interaction between the encoder and the transport layer, then the encoder can be informed about the network condition. Depending on that, the encoder can adjust its coding scheme. In the MBCPT coding scheme, if the buffer is getting full, that means that the bit generating rate is overwhelming the packetization rate and the encoder will switch to a coarse quantizer with fewer steps or loosens the threshold to decrease its output rate. In this way, smooth quality degradation is obtainable. However, this also complicates the encoder design.

It is possible to use the congestion control of the network protocols to prevent the drastic quality change by assigning different priorities to packets from different passes. Without identifying the importance of each packet and discarding packets blindly sometimes brings disaster and can cause a session shut down. For example, if the side information gets lost it can have a severe impact on the decoding process. In the MBCPT coding scheme, side information and packets from pass 1 are assigned highest priority and higher pass packets are assigned with decreasing priority.

D. Interaction with Protocols

In the ISO model, physical, datalink and network layers comprise the lower layers which form a network node. The higher layers have transport, session, presentation and application layers and typically reside in a customer's premises. The lower layers have to do nothing about the signal processing and only work as a "packet pipe". The physical layer requires adequate capacity and low bit-error-rate which are determined only by technology. The datalink layer can only deal with link-management because all the mechanics, like requesting retransmission, is not feasible in packet video transmission. The network layer has to maintain orderly transmission by deleting the delay jitter with input buffering. Otherwise, it can take care the network congestion by assigning transmission priority.

As the higher layers reside in the customer's premises, it performs all the functions of the packet video coder. The transport layer does the packetization and reassembly. The packet length can be fixed or variable. Fixed packet length simplifies segmentation and packet handling while a variable packet length can keep the packetization delay constant. The session layer supervises set-up and tear-down for sessions which have different types and quality. There is always a tradeoff between quality and cost. The quality of a set-up session can be determined by the threshold in the coding scheme and the priority assignment for transmission. Of course, the better the quality, the higher the cost. Fig. 13 shows the tradeoff between PSNR and video output rate by adjusting thresholds. The presentation layer does most of the signal processing, including separation and compression. Because it knows the video format exactly, if any error concealment is required, it will be performed here. The application layer works as a boundary between the user and the network and deals with all the analog-digital signal conversion.

VI. PERFORMANCE RESULTS

Results obtained in this packet video simulation show that substantial compression can be obtained while maintaining high image quality through the use of this differential MBCPT scheme. The monochrome sequence used in this simulation contains 16 frames, each of size 256x256 pixels with 8 bits per pixel, which results in a bit rate of 15.3 Mbits/s, given a video rate of 30 frames/s. As Table 2 shows, the average data rates of our system is 1.539 Mbits/s. The compression rate is about 10 with a mean PSNR of 38.74 dB where PSNR is defined as

$$\text{PSNR} = 10\log_{10}\frac{\sum(255)^2}{\sum(x_{i,j}-\hat{x}_{i,j})^2}$$

Fig. 14 shows the data rate of the sequence frames with side information, 4 passes and total rate. It is clear that the data rate of pass 1 is constant as long as the quantization mode remains the same. Side information and data from pass 2, even pass 3, is also. The data rate of pass 4 is bursty and highly uncorrelated. As pass 4 data is not essential to the reconstruction of the image, the rate profiles as shown in Figure 14 and Table 1 suggest the use of a reserved channel of some sort for passes 1–3 and the side information, and a perhaps more unreliable channel for pass 4 data which comprises more than 30% of the total traffic. Such a situation

can be accommodated in a variety of systems such as a token ring network or a circuit switched network with a packet-switched overlay.

Fig. 15 shows the PSNR for each frame in the sequence. Notice that the standard deviation of the PSNR is only 0.2 dB, which implies a substantial uniformity of quality, at least in terms of objective performance measures. If constancy with regards to some subjective criterion is desired, it would be necessary to incorporate this in the determination of the thresholds and the decision mechanism for the quad tree. In the simulation, the same threshold has been used throughout the sequence. If further flexibility, say for higher visual quality is desired, a varying threshold can be used for different frames. That may generate a more variable bit rate.

From the difference images of this sequence, frames 1-8 seem quite motionless while frames 9-13 contain substantial motion. We adjusted the traffic condition of the network to force some of the packets to get lost and thus check the robustness of the coding scheme. Heavy traffic was set up in the motionless and motion period separately. The average packet loss percentage was 3.3% which is considered high for most networks. Fig. 16 shows images which suffered packet losses from pass 4. As can be seen, the effect of lost packets is not at all severe, even if the lost packet rate is unrealistically high. This is because the performance from the first three passes is relatively good and the packet from the fourth pass is not essential for reconstruction. Fig. 17 shows the case when packet loss occurs in pass 1. Clearly there are visible defects in the motion period. What's worse, the error will propagate to the following frames. Apparently, the replenishing scheme used here is not sufficient in areas with motion. It is believed that this inconsistency can be eliminated with a motion compensator algorithm which would find the appropriate area for replenishment and error concealment which limits the propagation of error.

VII. CONCLUSIONS

The network simulator was used only as a channel in this simulation. In fact, before the real-time processor is built, a lot of statistics can be collected from the network simulator to improve upon the coding scheme. These include transmission delays and losses from various passes under different network loads. For resynchronization, the delay jitter between received packets can also be estimated from the simulation. The environment for tomorrow's telecommunication has been described and requires a flexibility which is not possible in a circuit-switched network. With all the requirements for applying packet video in mind, MBCPT has been investigated. It is found that MBCPT has appealing properties, like high compression rate with good visual performance, robustness to packet lost, tractable integration with network mechanics and simplicity in parallel implementation. Some additional considerations have been proposed for the entire packet video system, like designing protocols, packetization, error recovery and resynchronization. For fast moving scenes, the differential MBCPT scheme seems insufficient. Motion compensation, error concealment or even attaching function commands into the coding scheme are believed to be useful tools to improve the performance and will be the direction of future research.

VIII. REFERENCES

[1] W. Verbiest and L. Pinnoo, "A Variable Bit Rate Codec for Asynchronous Transfer Mode Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 761–770, June 1989.

[2] M. Ghanbari, "Two-Layer Coding of Video Signals for VBR Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 771–781, June 1989.

[3] J. Darragh and R. Baker, "Fixed Distortion Subband Coding of Images for Packet-Switched Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 789–800, June 1989.

[4] F. Kishino, K. Manabe, Y. Hayashi and H. Yasuda, "Variable Bit-Rate Coding of Video Signals for ATM Networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp.801–806, June 1989

[5] G. Karlsson and M. Vetterli, "Packet Video and Its Integration into Network Architecture," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 739–751, June 1989.

[6] M. C. Rost, "Data Compression Using Adaptive Transform Coding," Ph.D. dissertation, University of Nebraska-Lincoln, 1988.

[7] M. C. Rost and K. Sayood, "A Progressive Data Compression Scheme Based on Adaptive Transform Coding," *Proc. 31st Midwest Symposium on Circuits and Systems*, St.Louis, MO, Aug.1988. pp. 912-915.

[8] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, J. Robbins, "Performance Analysis of Statistical Multiplexing for Packet Video Sources," *Proc. of the Globecom-87*, Tokyo, Japan, Nov. 1987, pp. 47.8.1–10.

[9] K. Sloan, Jr. and S. Tanimoto, "Progressive Refinement of Raster Scan Images," *IEEE Trans. Comput.*, vol. COM-28, pp. 871-874, Nov. 1979.

[10] H. M. Dreizen, "Content-Driven Progressive Transmission of Grey-Scale Images," *IEEE Trans. Commun.*, vol. COM-35, pp. 289-296, March 1987.

[11] D. Vaisey and A. Gersho, "Variable Block-Size Coding," *Proc. ICASSP*, pp. 1051–1054, April 1987.

[12] Y. S. Ho and A. Gersho, "Variable-Rate Multi-Stage Vector Quantization for Image Coding" in *Proc. ICASSP*, vol. 2, pp. 1156-1159, 1988.

[13] D. Nelson, K. Sayood, G. Ankenman and H. Chang, "Pnetsim System Programmer's Manual," University of Nebraska-Lincoln, Final Report for ARMY Contract DAAB07-85-K-K535, Dec. 1986.

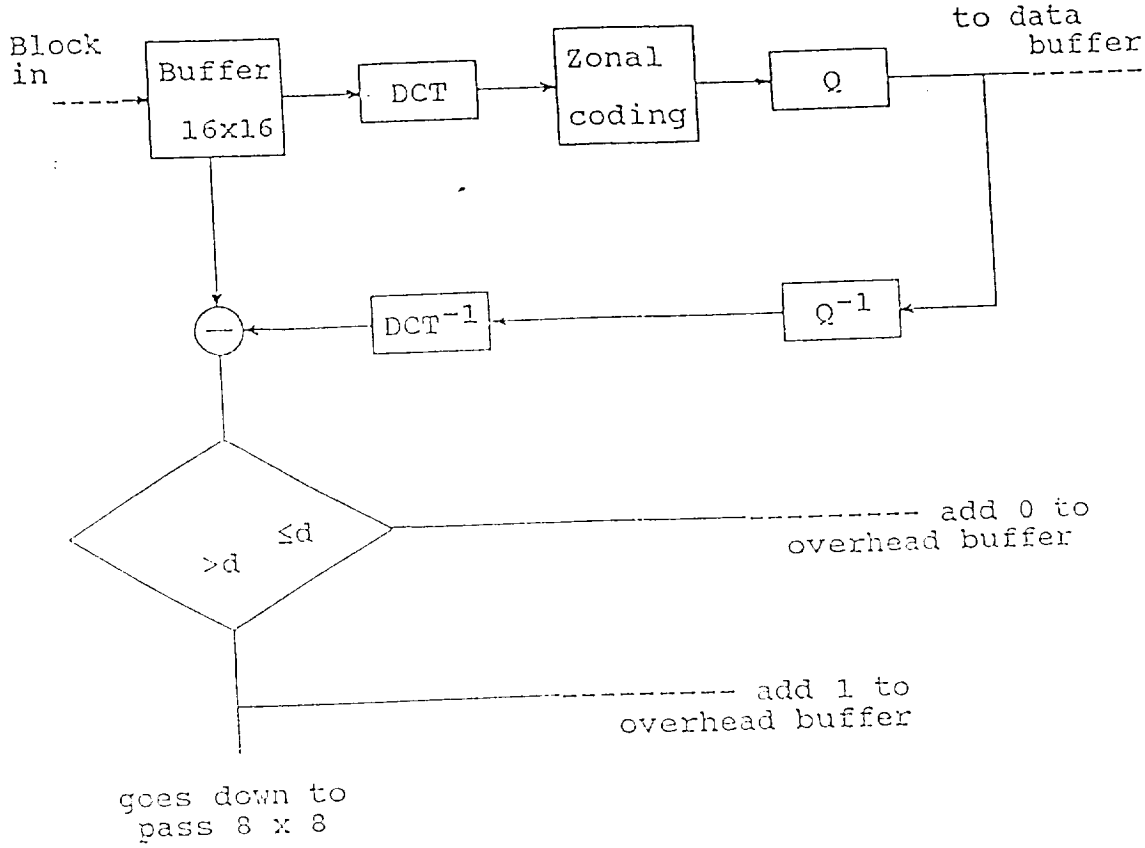


Figure 1 Structure of the first pass consisting of 16x16 blocks for MBCPT.

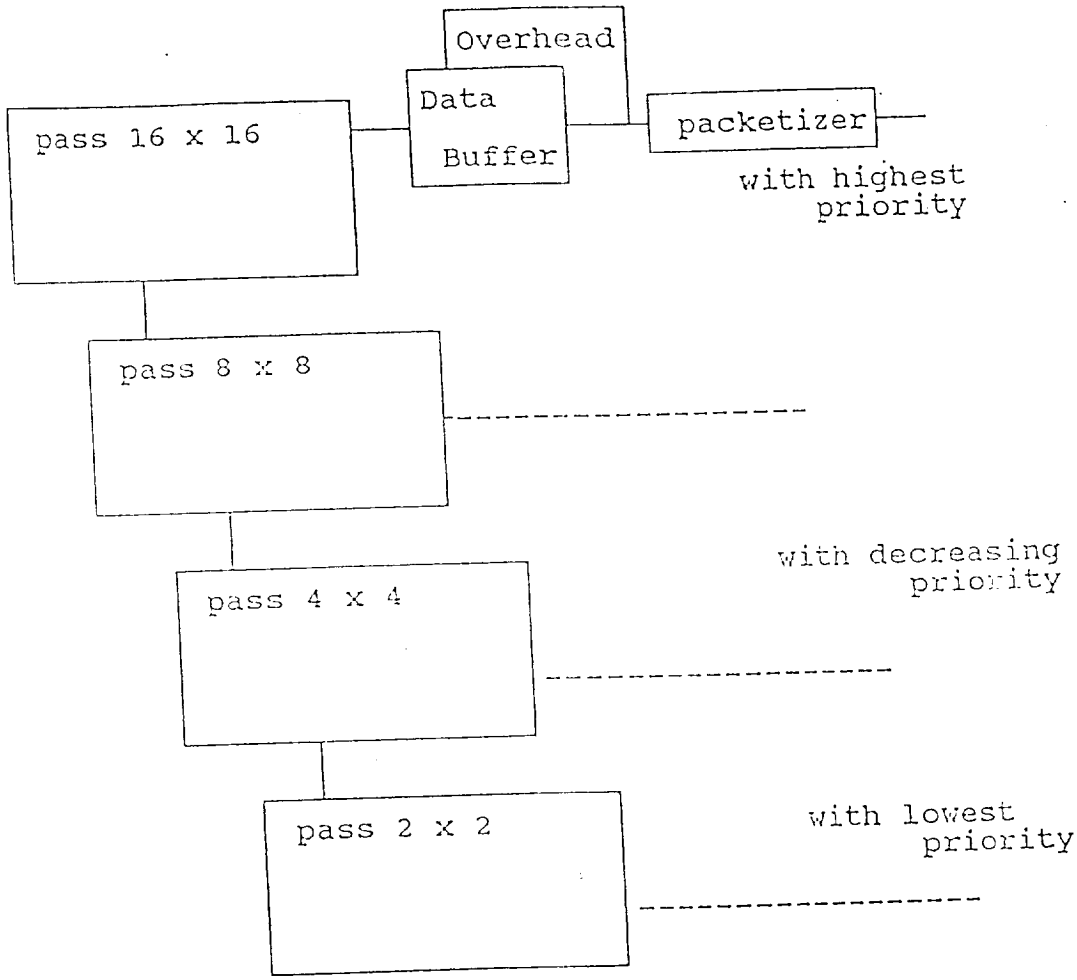


Figure 2 Parallel structure for MBCPT.

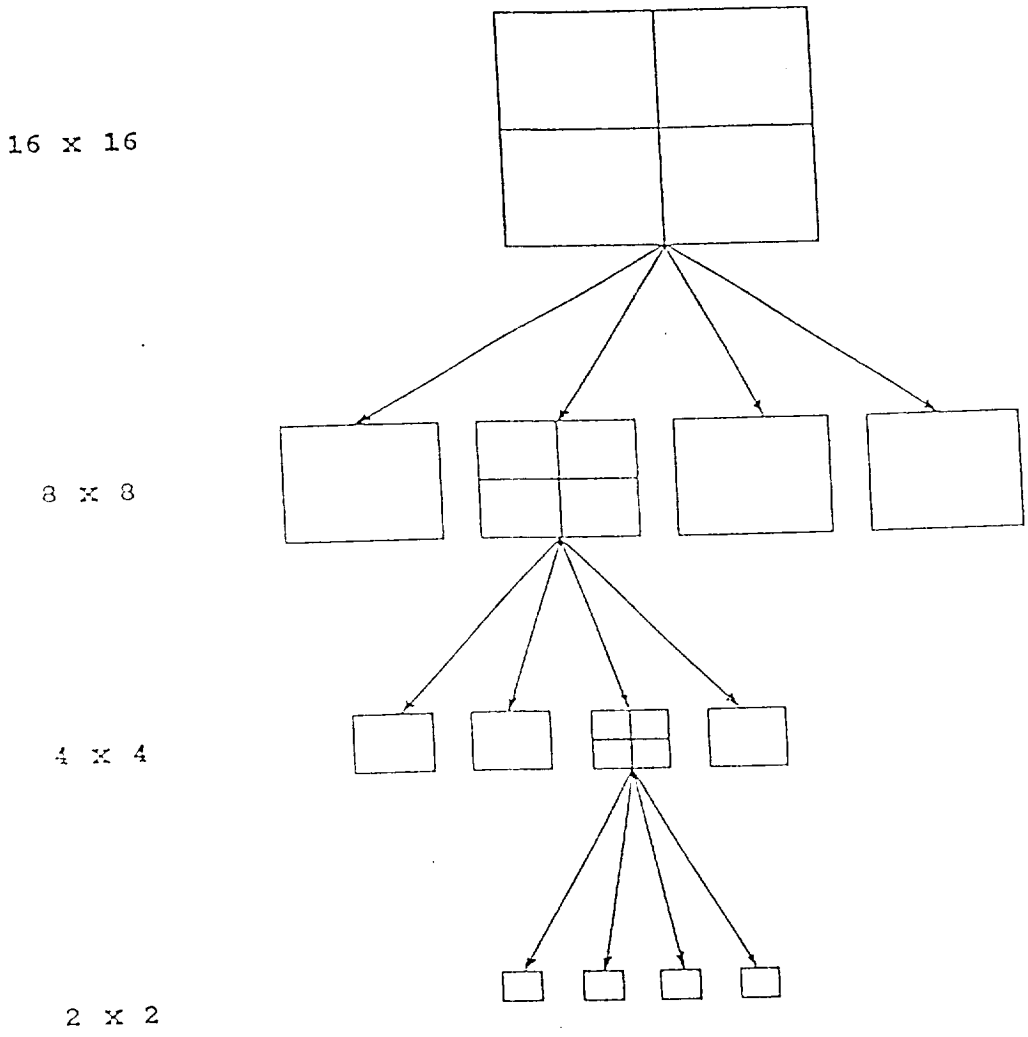
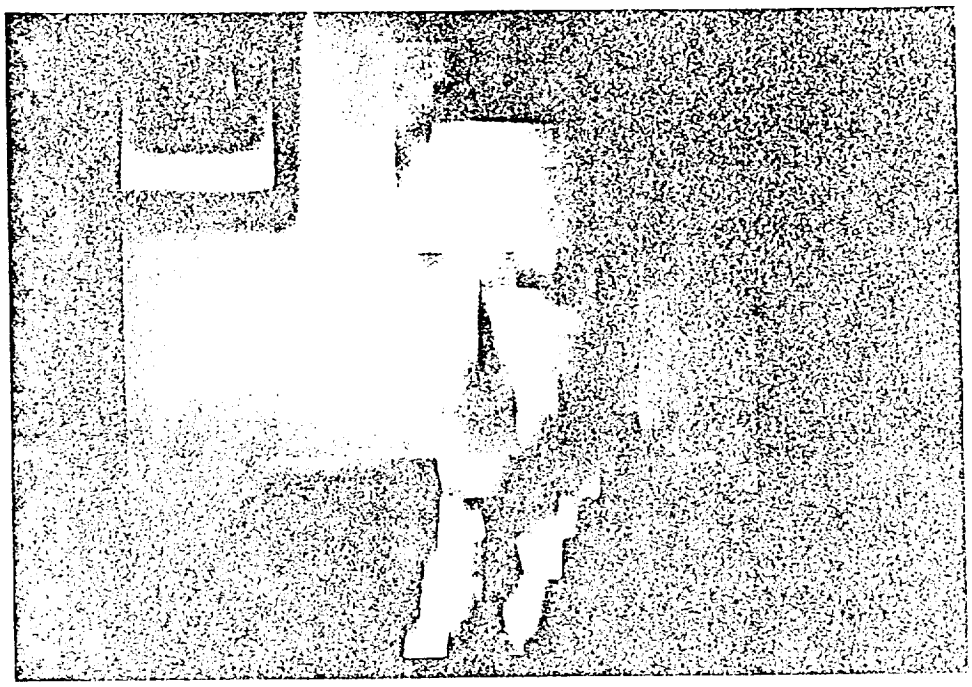
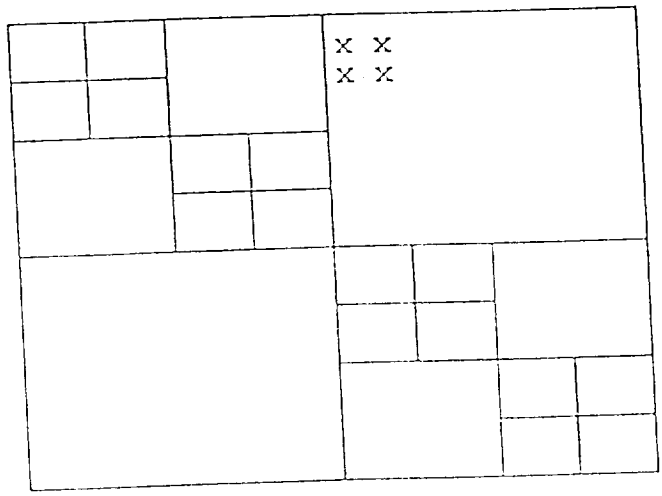


Figure 3 Example of the quad tree structure.

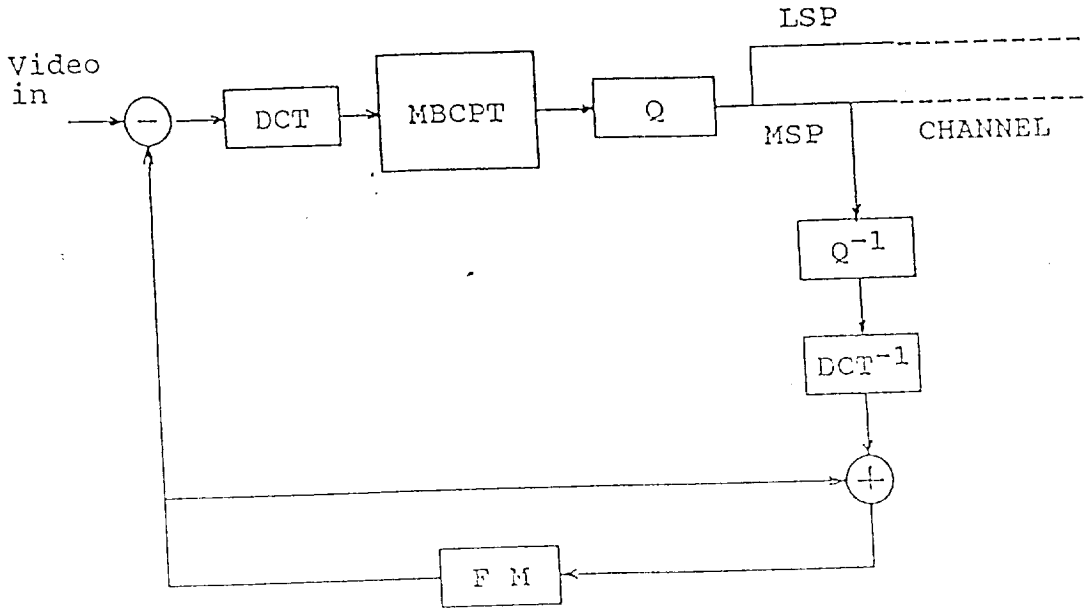




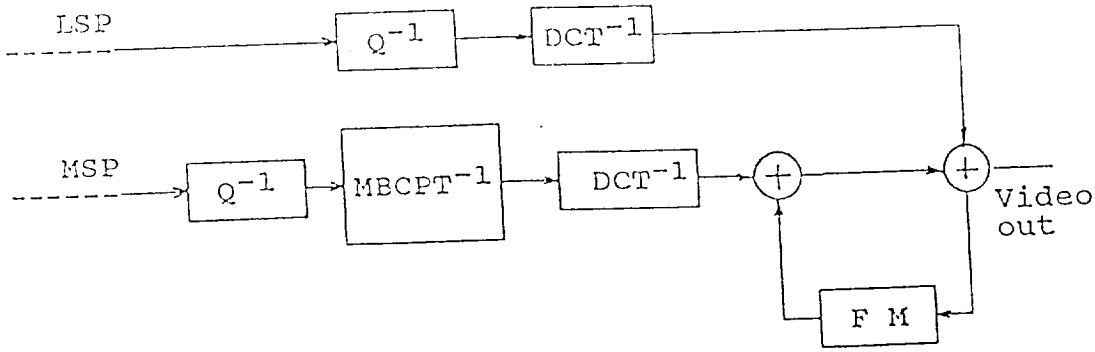
16 x 16

overhead = 1,1001,1001,1001,1001,1001

Figure 8 Overhead assignment and zonal coding.

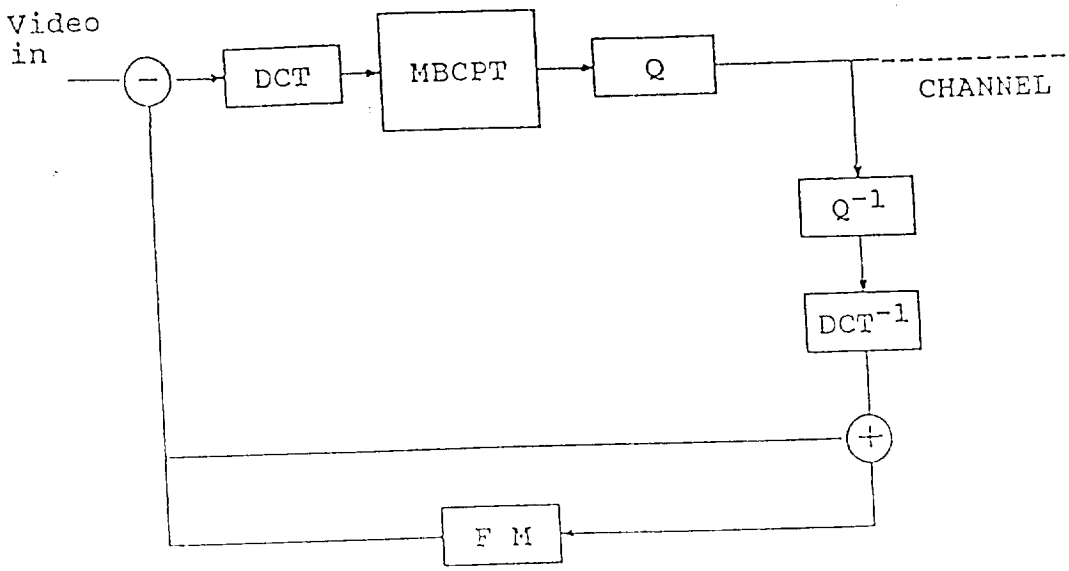


Coder

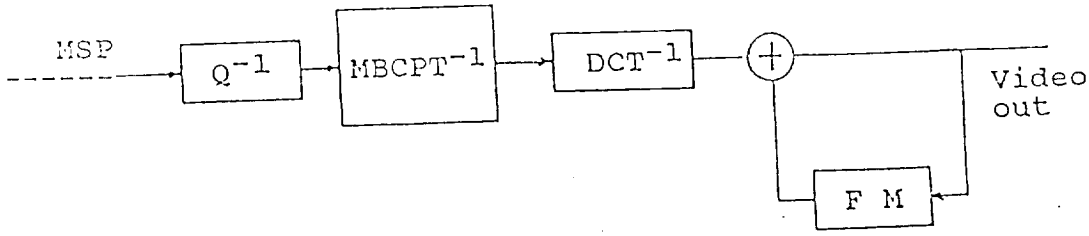


Decoder

Figure 9. Differential MBCPT coding scheme (1).



Coder



Decoder

Figure 10 Differential MBCPT coding scheme (2).

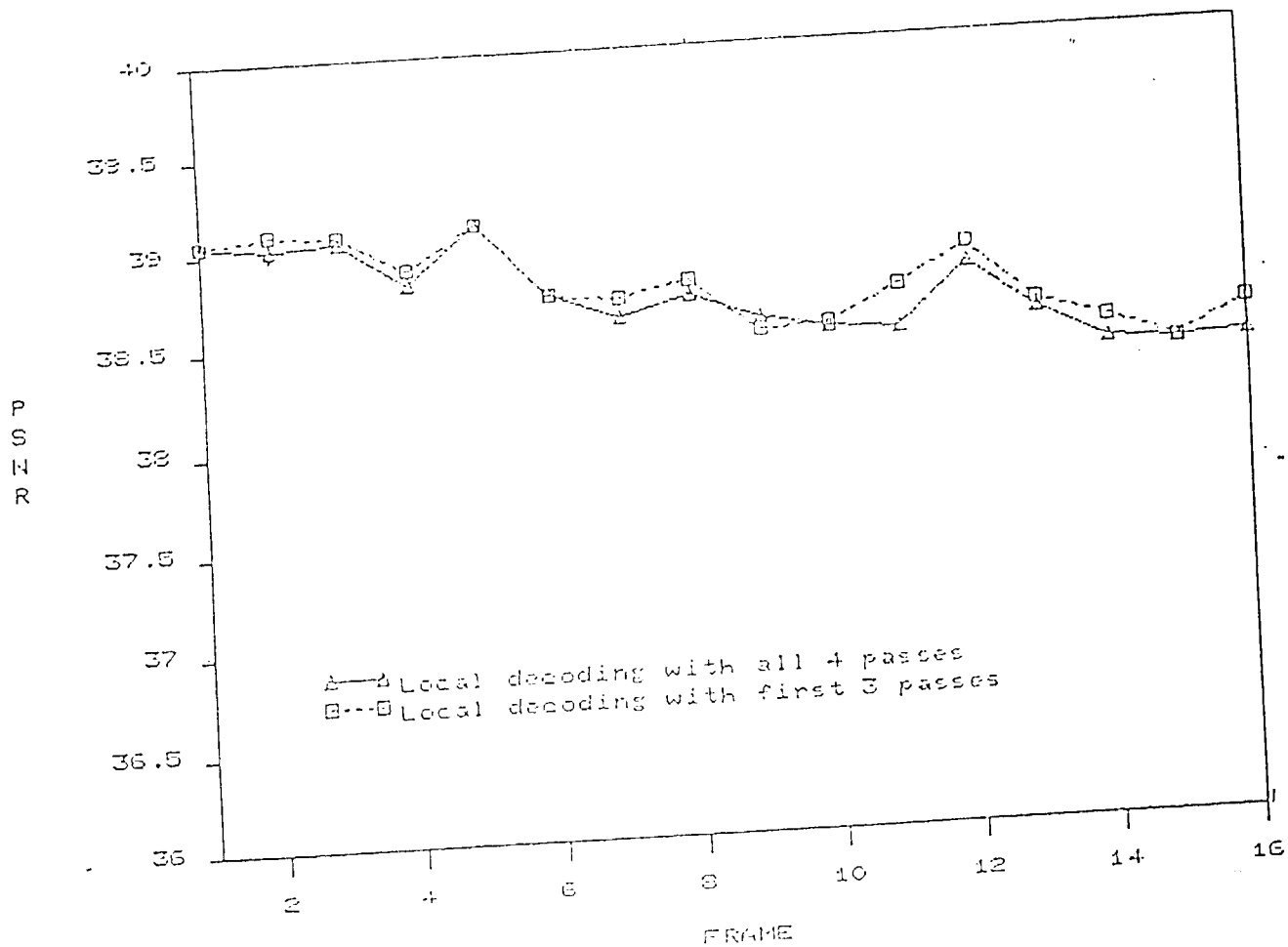


Figure 11 Performance of two differential MBCPT schemes without packet loss.

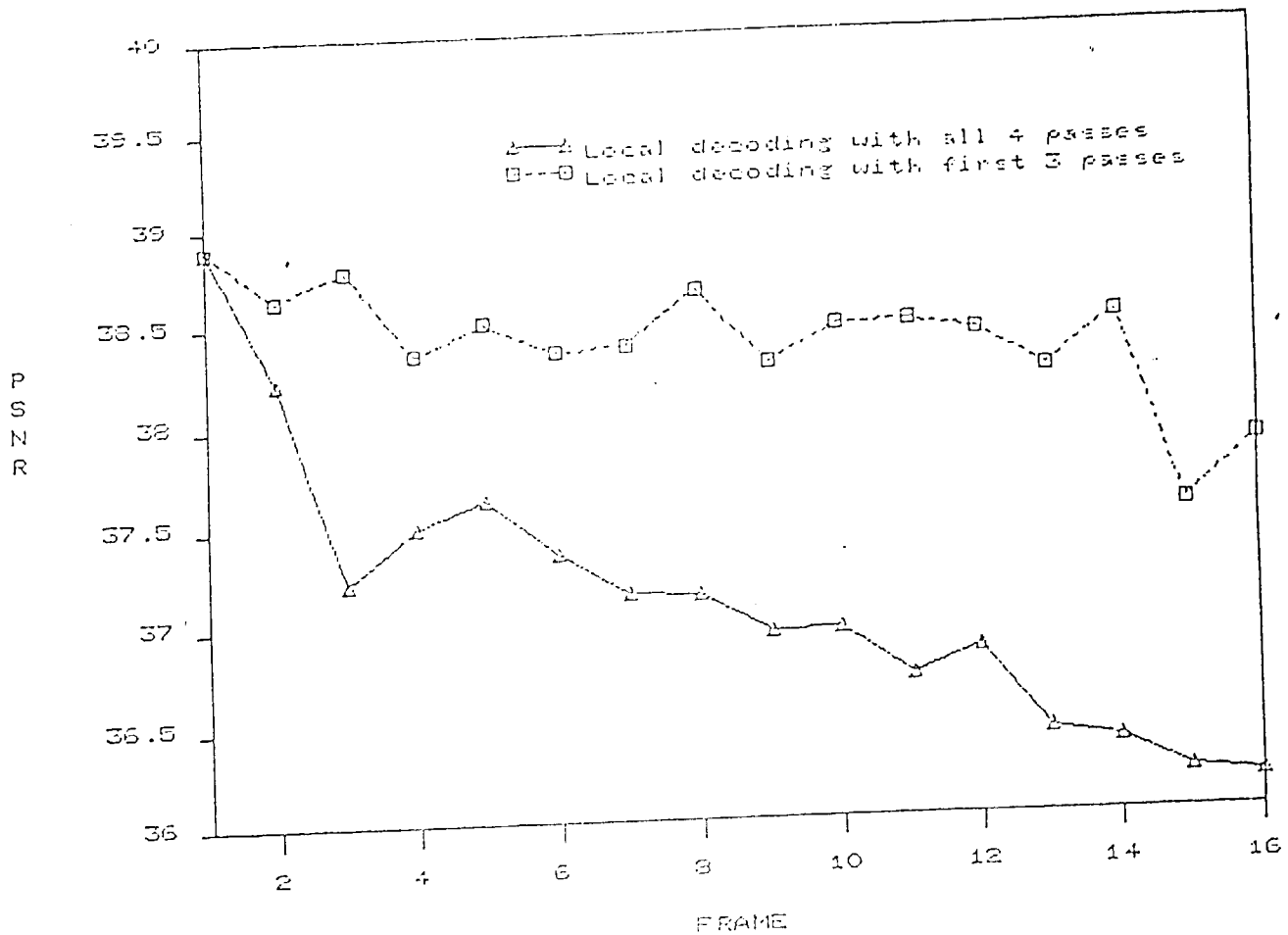


Figure 12 Performance of the two MBCPT schemes with packet losses from pass 4.

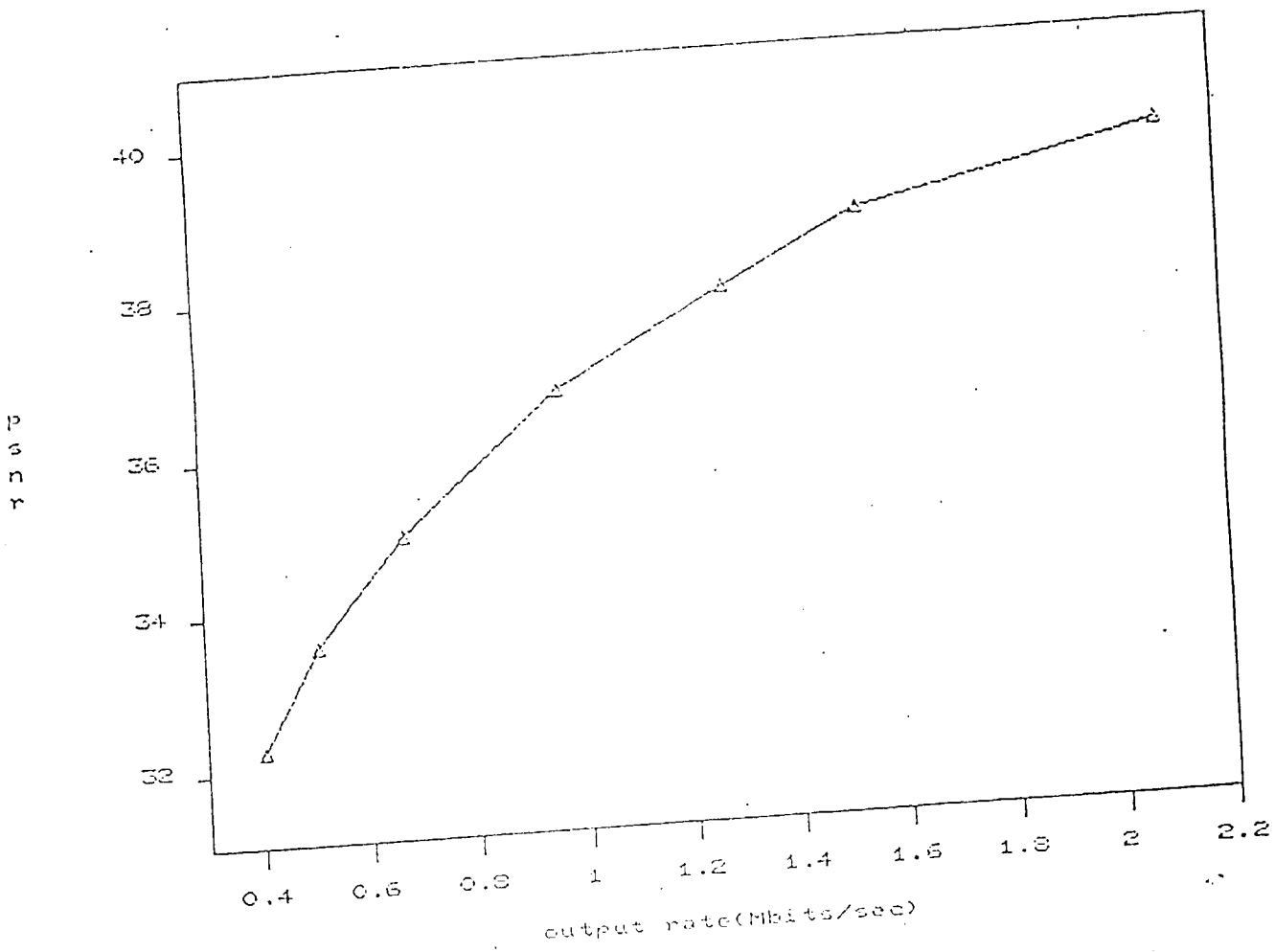


Figure 13 PSNR versus video output rate (video transmission at 30 frames per second).

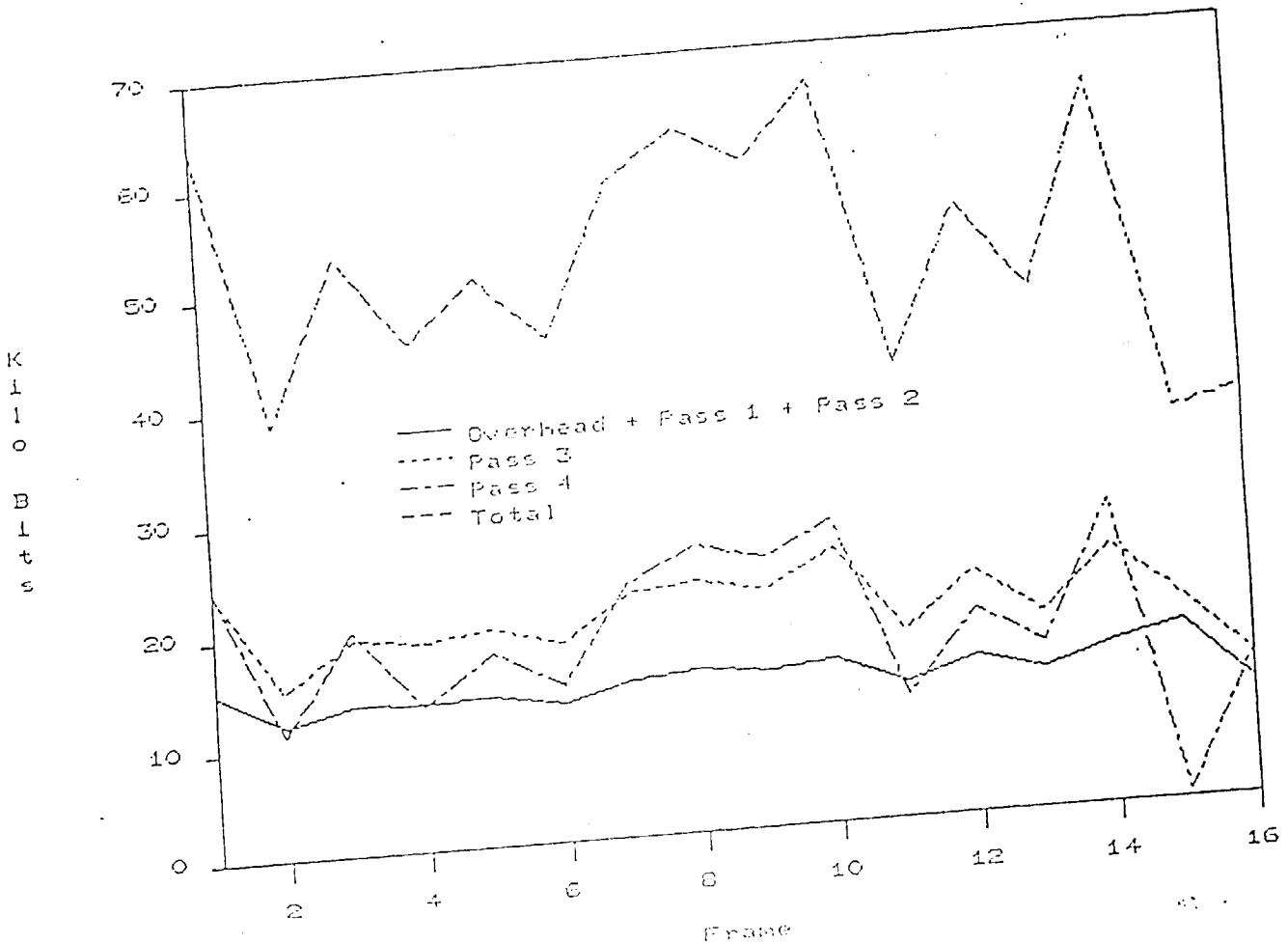


Figure 14 Data rate of simulation sequence frames.

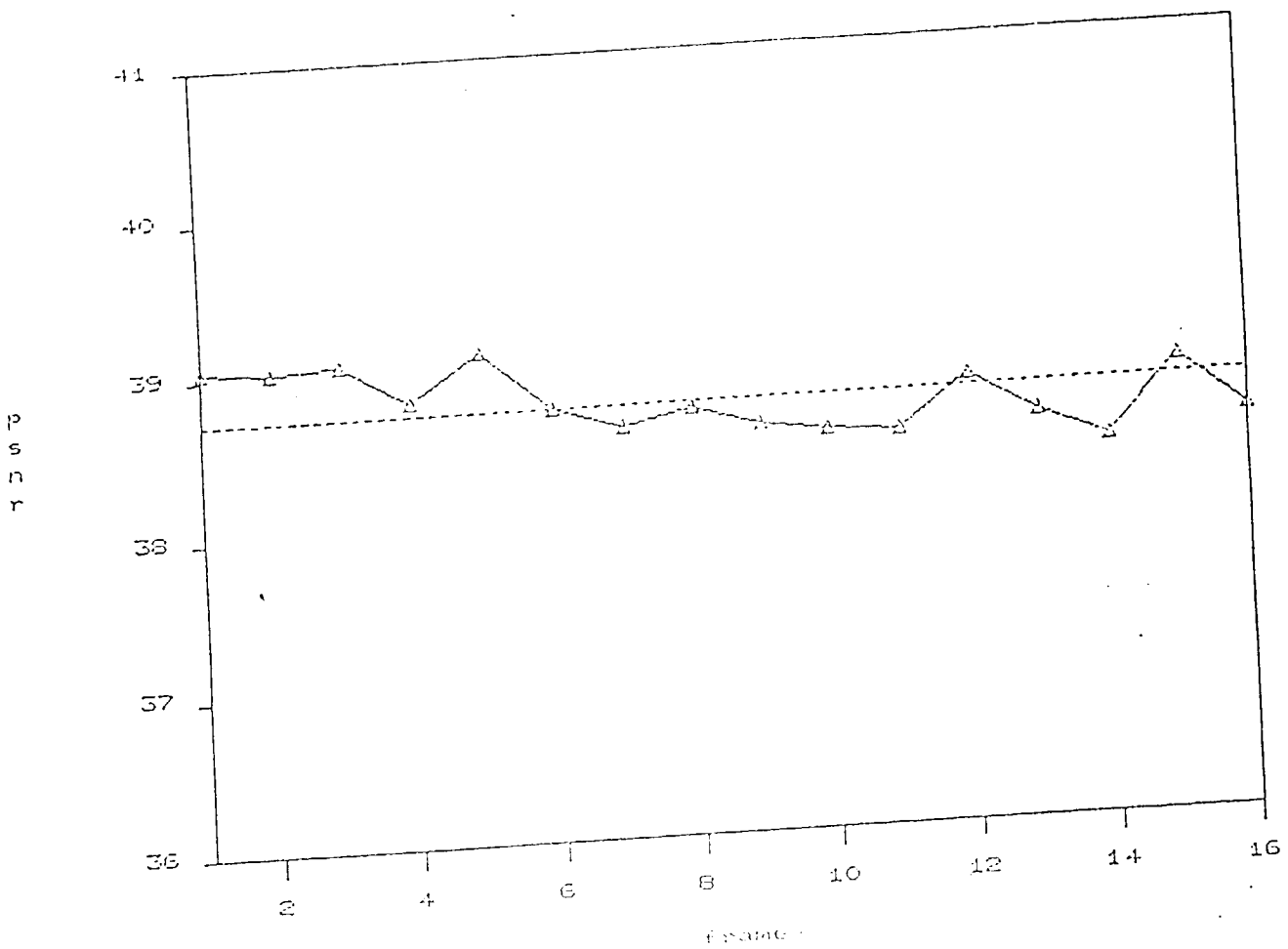
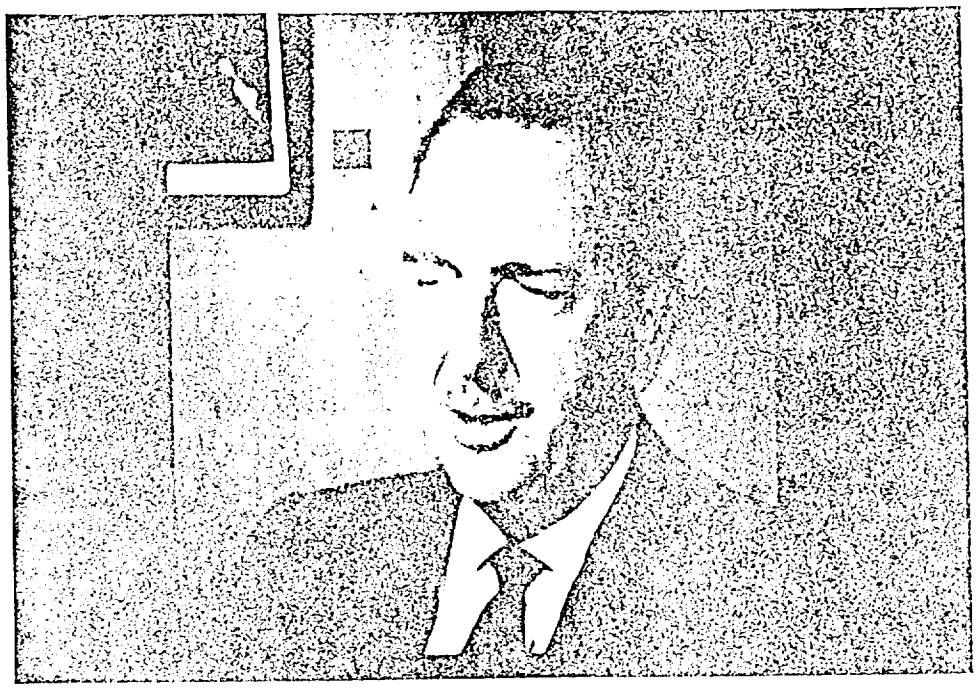


Figure 15 PSNR of simulation sequence frames.

The effect of pass 1 packet

The effect of pass 1 packet



Appendix 2- Item 11

S7-61

14029

P 4

A PROGRESSIVE DATA COMPRESSION SCHEME BASED UPON ADAPTIVE TRANSFORM CODING[1]

Mixture Block Coding of Natural Images

Martin C. Rost and Khalid Sayood
Department of Electrical Engineering
University of Nebraska, Lincoln, NE 68588 0511

N92-23423

Abstract

A method for efficiently coding natural images using a vector-quantized variable-blocksized transform source coder is presented. The method, Mixture Block Coding (MBC), incorporates variable-rate coding by using a mixture of discrete cosine transform (DCT) source coders. Which coders are selected to code any given image region is made through a threshold driven distortion criterion. In this paper, MBC is used in two different applications. The base method is concerned with single-pass low-rate image data compression. The second is a natural extension of the base method which allows for low-rate progressive transmission (PT). Since the base method adapts easily to progressive coding, it offers the aesthetic advantages of progressive coding without incorporating excessive channel overhead. Image compression rates of approximately 0.5 bit/pel are demenstrated for both monochrome and color images.

1 Introduction

Natural images contain regions of high and low detail. The regions of high detail are more difficult to code than those of low detail. Since the difference between number of bits required to code these two types of regions with acceptable distortion levels can be quite large, it is desirable to use a variable-rate source coding method. One way to do this is by using a transform coder which has more than one blocksize. Regions of high detail are coded with smaller blocks, while regions of low detail are coded with larger blocks. If a similar number of bits are used to code each of the different blocksizes, variable-rate coding is achieved. When trying to maximize the quality of the reconstructed image, better usage of coding bits can be attained with the use of vector quantization. The goal is to describe a method for coding images using both vector quantization and multiple-blocksize transform coding. A block-threshold technique is used to select the blocksize used to code any particular region of the image.

Both vector quantization and transform coding are block coding methods. Block coding methods are very useful when designing low-rate image compression systems, and are used almost exclusively when coding at low rates (<1bit/pel). Traditional methods presented in the literature use either of these techniques, but have rarely use both together until very recently. One of the earlier publication to do this appeared in 1984 [1].

When using traditional vector quantizers for coding images, small blocksizes are used to limit the size of the required quantizer codebook. But, when coding natural images with a very small number of bits it is desirable to use as large a blocksize as possible to take maximum advantage of the high inter-pixel correlations. In general, this blocksize is usually larger than vector quantization techniques can comfortably handle. One method to overcome this problem incorporates subsampled vector quantization [2], but little has been done to use traditional vector quantization with variable-rate coding.

Small codebooks are needed because the best performing vector quantization techniques (i.e., the LBG method [3]) are clustering techniques whose codebooks are very unstructured. As a result, the codebooks are difficult to construct and use. If a codebook is designed to be less computationally intensive, such as with lattice quantizers [4], or pyramid vector quantizers [5], the attainable distortion per codeword increases for a given coding rate.

Transform coding techniques easily allow for the use of large blocksizes so high data compression ratios can be attained. But, it is difficult to keep good high-detail resolution when using large transform blocksizes [6]. This is true since most methods code only the low-frequency high-energy transform coefficients [7]. As a result, the high-frequency coefficients are often ignored. Since these coefficients carrying most of the information about the image's finer detail image quality suffers. Even when the image coefficients are vector quantized so more coefficients can be coded for a given average rate, it can still be difficult to get good high-detail resolution.

By using more than one blocksize, some of the inherent problems associated with low-rate transform coding can be overcome. Especially when vector quantizers are used to code the transform coefficients. The vector quantizer codebooks used here are of low dimensionality to keep their implementation simple. This is done by limiting the number of coefficients coded within any given block. For the examples given below, the vector quantizers code at most three coefficients as a vector for monochrome images, and nine coefficients (three transform pels) for color images. Never more than four transform pels are coded within any block, no matter is size.

After a block is coded using these coefficient-limited transform coders the distortion is measured, to see if it meets a predetermined coding threshold. If a block codes poorly, it is divided into four smaller blocks and recoded until a distortion threshold is meet or the minimum blocksize is attained. Thus, keeping the overall image integrity high by using the smaller blocks to more intensely code the high-detail regions.

In section 2 the threshold driven MBC coding algorithm is discussed. Also, the required overhead sent to the receiver to describe the final block structure of the coder is presented. Section 3 is a presentation of the MBC progressive transmission (MBC/PT) modification. In sections 4 and 5 the transform coder and the vector quantizers used in the example are shown. And finally, several examples are presented.

2 The Threshold Driven Structure of MBC

As mentioned above, each block of the image is coded using only a small number of transform coefficients. The difference between the original image and the coded image block is measured, and if the difference does not fall below a predetermined threshold, the block is divided into four smaller blocks and recoded. A new threshold is then applied to see if any of these blocks need to be divided further. This divide-and-test algorithm is continued until the entire image is coded with distortion that is less than the block threshold levels or the smallest blocksize is reached.

The monochrome images are coded using the maximum absolute difference distortion measure,

$$d = \max_i |x_i - y_i|$$

where the range of i is taken over the image block being coded, and y_i is the coded value of pixel x_i. For color images the maximum mean square difference is used,

$$d = \max_i \sqrt{(x_i - y_i)^T (x_i - y_i)}/3$$

where y_i is the coded value of color pel x_i.

[1]This work was supported by the NASA Goddard Space Flight Center under grant NAG 5-916.

To demonstrate the MBC method consider the coding of an example image segment. In the following it is assumed the image is coded with a starting blocksize of 16×16 and a smallest blocksize of 2×2.

First, a 16×16 block is coded, using the DCT method described below, and the distortion level for the block is measured. If this distortion is greater than the predetermined maximum level for 16×16 blocks, dmin(16×16), the block is divided into four 8×8 blocks for additional coding. After each of the 8×8 blocks is coded their resulting distortion levels are compared with the 8×8 distortion level, dmin(8×8). This process is continued until the only image blocks not meeting their given distortion threshold are those of size 2×2. Since 2×2 is the smallest allowed blocksize, these blocks are transmitted de facto, making no further attempt to improve their distortion level.

Each 16×16 block can be completely coded, using all blocksizes, before moving onto the next 16×16 block or all the 16×16 blocks of the entire image can be coded before moving to the 8×8 blocks. For MBC, this sequence is immaterial. The later method allows one to develop the progressive technique introduced in the next section.

Consider the example coding of a 16×16 image block shown in Figure 1. For clarity in the following material, let the four sub-blocks of an arbitrary block be numbered as shown in Figure 2, and let the block distortion thresholds be:

$$\text{dmin}(16\times16)=12,$$
$$\text{dmin}(8\times8)=12 \text{ and}$$
$$\text{dmin}(4\times4)=10.$$

After coding this example d(16×16) is, say, 30. So this block must be divided and recoded as four 8×8 blocks. Let the four coded 8×8 blocks have distortion levels, as shown in Figure 2, of 10, 15, 4, and 6. Since one of these 8×8 blocks fail to met dmin(8×8), it must be divided and recoded as four 4×4 blocks. Letting the 4×4 distortions levels of this 8×8 blocks be 10, 8, 15, and 9, it can be seen that only one of the 4×4 blocks fails to met dmin(4×4). This block is recoded as four 2×2 blocks.

The above coding process is depicted as a quad tree structure in Figure 3. Here there is

one 16×16 block,
four 8×8 blocks,
four 4×4 blocks and
four 2×2 blocks,

for a total of 13 blocks of coded information which are connected together through the use of a relative pointer scheme. To guarantee the receiver reconstructs the image correctly, a bit map of side information is sent along with the block coefficient information. One bit of side information is needed for each block of size greater than 2×2. If a block is to be divided, its bit value is set to 1; if not, its bit value is set to 0. To tell the receiver the ordering of these 13 coded blocks, 9 bits of side information are needed:

1,0,1,0,0,0,1,0,0.

The first bit shows the 16×16 block is divided into 8×8 blocks. The next four bits indicate only the second 8×8 block is divided. The last four bits indicate that only the third 4×4 block is divided into 2×2 blocks. Notice the 2×2 blocks of this example are placed with the bit maps generated at the 4×4 level, so no side information is needed to code them.

If b_i is the number of pels and d_i is the number of bits used to code a block on the ith pass, then the average coding rate for the blocks of this pass is $r_i = d_i/b_i$. In the last pass (n-th) there are no overhead bits, so $r_n=d_n/b_n$. The average coding rate for the entire MBC system is

$$R=\sum_i p_i r_i, \text{ and } \sum_i p_i=1. \qquad (1a,b)$$

where p_i is image fraction coded with i-th pass blocksize.

There is no reason to force the coding method of one blocksize onto another blocksize, as is the case for the examples of this paper. Sometimes this maybe detrimental or, in fact, be impossible. Consider the case where blocks of size 16×16 are code with, say, six DCT coefficients. The 8×8 and 4×4 blocks could be coded in a similar fashion but, of course, it is impossible to DCT code the a 2×2 blocks with six transform coefficients. This indicates the 2×2 blocks must be coded with a different method. Likewise, as indicated above, there is no reason to keep the distortion threshold levels constant. It is even possible to change the distortion measure for each blocksize. And, since the different blocksizes encompass different spatial frequency ranges it may be desirable to do this.

3 Progressive Transmission MBC

Progressive transmission has grown out of need to transmit images over channels whose bandwidth is dramatically smaller than what is available for the timely reconstruction of full-field imagery. Since slow-scan reception of images is nonaesthetic, the need to update the image on a frame-by-frame basis (or progressive basis) has arisen. In general, most methods found in the literature are concerned with perfect reconstruction of the image. The image is transmitted on a continually improving basis until it is completely transmitted so that it can be reconstructed without error. This requires the transmission of far more data than is needed to attain a visually pleasing reconstruction of the image (as is the case considered here). But, much of this literature is directly applicable to the low-rate transmission of images since almost every PT method reconstructs a visually acceptable field within a limited number of passes [e.g., 12].

In the previous section a single an example 16×16 block was coded by passing through all of the necessary blocksizes before moving on to the next 16×16 block. But, if the entire image is passed through for each blocksize and the difference image is save for additional coding as is necessary in the next pass, the MBC method can be used as a PT coder. If each pass is immediately transmitted, the receiver can be reconstructing a crude representation of the image using these larger blocksize coefficients while the coder is processing the next pass. All passes after the first need only code the image residuals. The residuals coding information received in subsequent passes is added to the "already waiting" image of the receiver. The image is updated using smaller blocks so it acquires more clarity with each pass. Since these blocks are of smaller size, each pass updates higher-frequency image components than were coded in the previous passes.

Since the first pass is coded with very few bits the receiver has an image, although a crude image, almost immediately. Since the successive passes are coding the difference image instead of the original there is no serious problem in updating the base MBC method for PT coding.

Since only image regions that code poorly are updated in successive passes, only these regions of the image which need additional coding will continue to use coder resources. This means the regions of the image with low detail are coded quickly, and remain fixed, as the rest of the image continues to change as the information for each new pass is received.

The high detail regions of an image are coded more than once when using the MBC/PT method. Not only do the high detail regions require a greater channel capacity to transmit their coefficients because smaller blocksizes are being used but, it they also require channel resources in each of the previous passes. The rate for a MBC/PT coder is calculated using (1a), but the pass fractions are no larger constrained to add to one, in fact

$$\sum_i p_i \geq 1, \text{ and } p_i \geq p_j \text{ for } i \leq j. \qquad (2)$$

Thus, applying this to (1a) shows that is it possible to have $R_{MBC/PT} \geq R_{MBC}$. But this can be offset by the fact that MBC/PT may converge to the original image more quickly and require fewer blocks since the busy sections of the image are coded with information that is taken from one than a single pass. As is shown in section 5, an MBC/PT image can require fewer coding bits to transmit than image of similar quality using the MBC method.

4 The Transform Coder

If a large block does not adequately code a given image region, it is divided into smaller blocks and recoded. So there is no strict advantage in using a large number of coefficients to code any particular blocksize. In fact, there is a tradeoff between expending more effort coding the larger blocks so fewer smaller blocks are used, and coding the larger blocks minimally so to let the threshold algorithm assign more smaller blocks for coding.

For the examples of this paper it was chosen to code each block with only four DCT transform coefficients, including the dc and three lowest order frequency coefficients (Figure 4). This was done, not so much to attain the best overall coding rate, but, to strike a median between PT coders which code a minimum of information about a given block [8] and those which code a large amount of information per block [9]. This accomplishes two things. Firstly, it shows an image can be adequately coded in a relatively small number of passes (four for the examples here) using a small number of transform coefficients at each pass, and secondly, it shows that this can be done using a simple coding algorithm for each pass. In addition, when using the same transform coder for each pass it is also possible to use share the same vector quantizer between all of the passes. This saves quantizer design effort.

5 The Quantizers

The following four paragraphs describe the quantizers used to code the examples discussed in section 5 of this paper. The remaining paragraphs discuss the details of LBG quantizers used.

When the MBC method was used to code monochrome images, the dc transform coefficients were coded with an 8-bit linear scalar quantizer (LSQ), and the ac transform coefficients were coded as a single 3-dimensional vector using an LBG vector quantizer whose codebook was of size 256.

When the MBC/PT method was used to code monochrome images, the quantizers for the first pass were different than those of subsequent passes. The these later passes code difference images that have nearly zero means blocks, while the first pass deals with the original image which is not zero mean. The dc coefficients for the first pass were coded with an 8-bit LSQ. In subsequent passes, the dc coefficients were coded with a 5-bit optimal laplacian scalar quantizer (OLSQ) [10]. The non-dc coefficients were coded the same for all passes using an LBG vector quantizer whose codebook contained 256 vectors.

The three non-dc coefficients of the YIQ images, when using the method MBC, were quantized with a vector quantizer whose codebook contained 1024 vectors. The dc Y-components were quantized with an 8-bit LSQ and the dc I- and Q- components were coded with a 5-bit OLSQ.

As with the monochrome MBC/PT method, the YIQ MBC/PT images where coded differently in the first pass than in the subsequent passes. As was the case for MBC, the dc Y-component was quantized with an 8-bit LSQ, while the I- and Q-components were quantized with a 5-bit OLSQ. In subsequent passes, the dc coefficients were quantized with an LBG vector quantizer whose codebook contained 64 vectors. The non-dc coefficients were quantized with the same LBG vector quantizer, whose codebook contained 1024 vectors, for all passes.

It was decided to use the same vector quantizer for the ac transform coefficients of each pass, no matter the blocksize. This rule was chosen because the ac coefficients for all passes were found to be distributed in a similar fashion. The only adjustment to be made was to allow for the differences in the coefficient variances, which had to be adjusted for each pass to guarantee a universal match the codebook variance.

By using this variance matching technique, more training vectors could be taken from a single image, so fewer images were needed to develop an adequate training set. The training set was built using scaling factors that mapped the unquantized ac coefficients of each pass into unit pdf. The scaling factors used are shown in Tables 1 and 2. The final vector quantizer codebooks, as was indicated above, hold 1024 9-dimensional vectors for the YIQ coders and 256 3-dimensional vectors for the monochrome coders. They were built using a training set taken from three images different from the one coded for this paper.

The final vector quantizers were chosen to be nonadaptive. This is, the codebook was not modified to more effectively quantize out-of-bound source vectors as the coder moves from image to image [e.g., 11]. This was done for two reasons. Sometimes the overhead required to implement such a technique can be overly expensive and the return acquired from it minimal. The extra effort needed stood against the design goal to construct a "simple to implement" vector-quantized adaptive transform source coder.

6 Results

All of the examples, as listed in Tables 3-6, use the 512×512 RGB woman/hat picture of the UCLA database. The monochrome examples use the green (G) color field, while the YIQ images are made using the RGB to YIQ transformation matrix of [12]. All the examples use a starting blocksize of 16×16 and a final blocksize of 2×2. These tables list the number of blocks coded for each blocksize, and the thresholds used to test the quality of the coding passes. Also, the MBC/PT tables list the average coding rate that has accumulated after each pass.

This rate is based upon the average of the coding bits as spread across the entire image, without concern for what fraction of the image is coded within any particular pass. These rates represent the coding rate that is required to code the image if the coder where to stop with that particular pass. Since the remaining passes are yet to be coded, the image percentage coded within the indicated pass must be updated to include the image percentages coded in all of subsequent passes. For example, consider the MBC/PT rate of Table 4 when stopping at 4×4 blocks. In this case, the rate is .358 bits/pel, and the percentage of blocks coded with 4×4 blocks is 25.25 percent.

To code the monochrome image with MBC/PT only requires an extra .011 bits/pel over MBC. The overhead needed to code any given image is a function of the LBG codebook, and the codebook is a function of the training set used. A differently constructed codebook could offer different results. It is interesting to note that the YIQ MBC/PT method requires less coding rate to obtain the same image quality (PSNR) as is obtained when using MBC alone. It is clear that MBC/PT does not require excessive overhead to add the desirable PT feature.

References

[1] K. Sayood, J. Gibson and M. Rost, "An algorithm for uniform vector quantization design," *IEEE Trans. Inform. Theory*, vol. IT-30, no. 6, pp. 805-814, Nov. 1984.

[2] H.-M. Hang and B. G. Haskell, "Interpolative vector quantization of color images," *IEEE Trans. Commun.*, vol. COM-36, pp. 465-470, Apr. 1988.

[3] Y. Linde, A. Buzo and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84-95, Jan. 1980.

[4] J. D. Gibson and K. Sayood, "Lattice quantization," to appear in *Advances in Electronics and Electron Physics*.

[5] T. R. Fischer, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 568-583, July 1986.

[6] D. J. Vaisey and A. Gersho, "Variable block-size image coding," *Proc. ICASSP 87*, vol. 2, pp. 1051-1054, Apr. 1987.

[7] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*, Englewood Cliffs, NJ: Prentice-Hall, 1984.

[8] K. Knowlton, "Progressive transmission of grey-scale and binary pictures by simple, efficient, and lossless encoding schemes," *Proc. IEEE*, vol. 68, pp. 885-896, July 1980.

[9] E. DuBois and J. L. Moncet, "Encoding and progressive transmission of still pictures in NTSC composite format using transform domain methods," *IEEE Trans. Commun.*, vol. COM-34, pp. 310-319, Mar. 1986.

[10] W. C. Adams and C. E. Gissler, "Quantizing characteristics for signals having laplacian amplitude probability density functions," *IEEE Trans. Commun.*, vol. COM-26, pp. 1295-1297, Aug. 1978.

[11] D. J. Vaisey and A. Gersho, "Variable blocksize image coding," *Proc. ICASSP*, Apr. 1987, pp. 1051-1054.

[12] L. Wang and M. Goldberg, "Progressive image transmission by transform coefficient residual error quantization," *IEEE Trans. Commun.*, vol. 36, pp. 75-87, Jan. 1988.

[13] W. K. Pratt, *Digital Image Processing*, New York, NY: Wiley, 1978.

Table 1.
Vector quantizer scale factors for monochrome images.

Blocksize	scale factor
16×16	60
8×8	35
4×4	20
2×2	10

Table 2.
Vector quantizer scale factors for YIQ images.

Blocksize	scale factor
16×16	80
8×8	35
4×4	20
2×2	10

Table 3.
Monchrome MBC rate (bits/pel)

min blocksize	#blocks	%image	PSNR	rate	threshold
16×16	533	52.05	23.11 dB	–	33
8×8	1048	25.59	26.22	–	33
4×4	2950	18.01	29.61	–	33
2×2	2856	4.36	31.01	.479	–

Table 4.
Monchrome MBC/PT rate (bits/pel)

min blocksize	#blocks	%image	PSNR	rate	threshold
16×16	1024	100.00	23.13 dB	.063	33
8×8	1940	47.48	26.42	.170	33
4×4	3520	21.48	29.76	.368	33
2×2	2472	3.77	30.94	.490	–

Table 5.
YIQ MBC rate (bits/pel)

min blocksize	#blocks	%image	PSNR	rate	threshold
16×16	530	57.62	24.09 dB	–	12
8×8	976	23.53	26.77	–	12
4×4	2444	14.92	29.06	–	11
2×2	2854	3.94	29.69	.691	–

Table 6.
YIQ MBC/PT rate (bits/pel)

min blocksize	#blocks	%image	PSNR	rate	threshold
16×16	1024	100.00	24.12 dB	.110	12
8×8	1770	42.97	26.89	.220	12
4×4	2920	17.82	29.09	.402	11
2×2	1950	2.78	29.63	.516	–

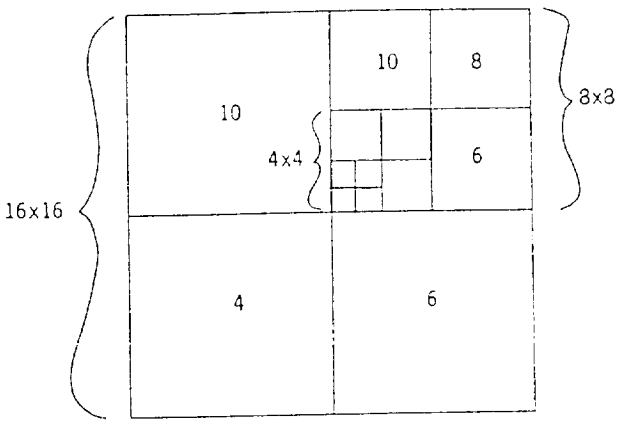


Figure 1. Example coded 16x16 block

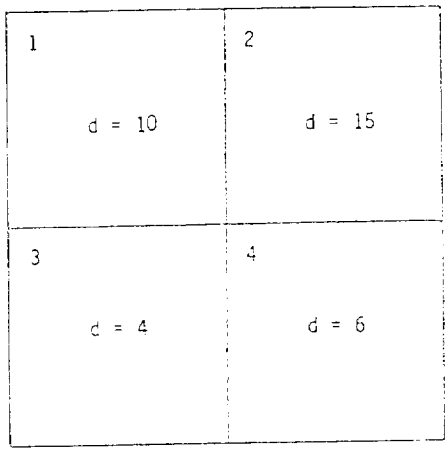


Figure 2

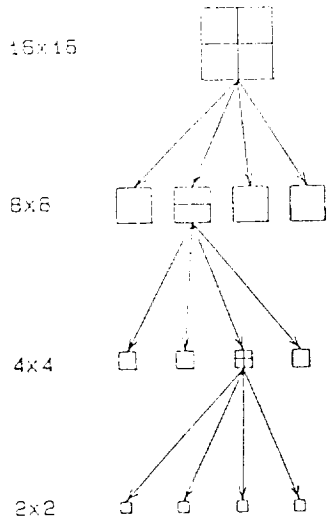


Figure 3. Example Tree Structure

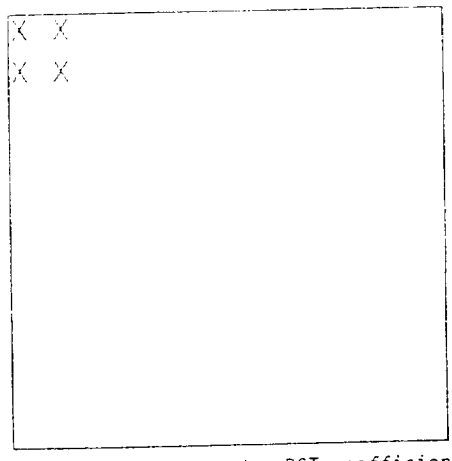


Figure 4. Lowest order DCT coefficients